Service Replication in Wireless Mobile Ad Hoc Networks

Dissertation

zur Erlangung des akademischen Grades
Doktor-Ingenieur (Dr.-Ing.)

vorgelegt dem Rat der Fakultät für Mathematik und Informatik
der Friedrich-Schiller-Universität Jena

von M.Sc. Mohamed Hamdy Mohamed El-Eliemy
geboren am 28.02.1979 in Kairo

Gutachter

1. **Prof. Dr. Birgitta König-Ries**
   Friedrich-Schiller-Universität Jena
2. **Prof. Dr. Heinz Schweppe**
   Freie Universität Berlin

Tag der öffentlichen Verteidigung: 14.12.2010

Ehrenwörtliche Erklärung

Hiermit erkläre ich,

- dass mir die Promotionsordnung der Fakultät bekannt ist,
- dass ich die Dissertation selbst angefertigt habe, keine Textabschnitte oder Ergebnisse eines Dritten oder eigenen Prüfungsarbeiten ohne Kennzeichnung übernommen und alle von mir benutzten Hilfsmittel, persönliche Mitteilungen und Quellen in meiner Arbeit angegeben habe,
- dass ich die Hilfe eines Promotionsberaters nicht in Anspruch genommen habe und dass Dritte weder unmittelbar noch mittelbar geldwerte Leistungen von mir für Arbeiten erhalten haben, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen,
- dass ich die Dissertation noch nicht als Prüfungsarbeit für eine staatliche oder andere wissenschaftliche Prüfung eingereicht habe.

Bei der Auswahl und Auswertung des Materials sowie bei der Herstellung des Manuskripts haben mich folgende Personen unterstützt:

- Prof. Dr. Birgitta König-Ries

Ich habe die gleiche, eine in wesentlichen Teilen ähnliche bzw. eine andere Abhandlung bereits bei einer anderen Hochschule als Dissertation eingereicht: Ja / Nein.

Jena, 19.10.2010

[Mohamed Hamdy Mohamed El-Eliemy]

Acknowledgments

First of all, I would like to thank Prof. Dr. Birgitta König-Ries for having welcomed me to join her research group, for supervising my work, for her continuous support and understanding, and also for her patience.

I would like to thank all the group members and all my colleagues, Uli, Fedor-chik, Torsten, Aygul, and Friederike, for their help and constructive discussions. With all of them, I have great memories and I wish them the best. I will never forget their support.

I would like also to thank Dr. Abdelouahid Derhab from the Department of Computer Engineering, CERIST Center of Research, Algiers, Algeria for his valuable comments and cooperation.

I would like to give special thanks to my wife for her support and encouragement. I dedicate this work to my wife Doaa, my daughters Toqa and Salma, my sister Aisha and my brother Ahmed, and the soul of my parents.

Deutsche Zusammenfassung

Die vorliegende Arbeit beschäftigt sich mit dem Management von Diensten in mobilen ad-hoc Netzwerken (MANETs). MANETs sind drahtlose Netzverbünde mobiler Einheiten die sich dezentral ohne eine übergeordnete Organisation selbst verwalten. Die Netztopologie eines MANET verändert sich dabei dynamisch mit der Bewegung der autonomen Teilnehmer. Sensor Netzwerke, Personal Area Networks und Satelliten Netzwerke sind typische Beispiele für derartige MANETs. Mit der wachsenden Bedeutung der drahtlosen Vernetzung mobiler Geräte haben sich MANETs in den vergangenen Jahren zu einem wichtigen Forschungsgebiet entwickelt. Im Katastrophenmanagement, bei zivilen Rettungsfällen oder in militärischen Szenarien kann ihre infrastrukturlose Selbstorganisation MANETs zum einzig möglichen Kommunikationsmittel machen.

Die mobilen Knoten eines MANETs kooperieren um essenzielle Netzwerkdienste wie das Routing und den Datentransport gemeinschaftlich zu gewährleisten. Resourcen wie die Bandbreite zwischen Knoten, die Rechenleistung der mobilen Geräte und ihre Batterieleistung sind dabei typischerweise stark begrenzt und zudem wechselnd. Das Teilen der verfügbaren Resourcen ist daher eine Notwendigkeit für das effiziente Funktionieren eines MANET. Dienstorientierte Architekturen (SOAs) stellen ein geeignetes Paradigma da, um geteilte Resourcen zu verwalten. Wenn verfügbare Resourcen als Dienst aufgefasst werden, lässt sich ihre Nutzung als Dienstanfrage bearbeiten. In diesem Zusammenhang ermöglichen SOAs Abstraktion, Kapselung, loose Koppelung, Auffindbarkeit von Resourcen und die für MANETs essenzielle Autonomie. Die Anwendung von SOAs auf MANETs findet daher zunehmend Beachtung in der Forschung.

Die wahrscheinlich größte Herausforderung für die Realisierung einer SOA in einem MANET ist die Gewährleistung einer hohen Verfügbarkeit der bereitgestellten Dienste. Ein Dienstanbieter kann temporär oder permanent unerreichbar werden wenn das mobile Netz wegen der Bewegung der Teilnehmer partitioniert. Darüber hinaus kann die Resourcenbeschränkung eines einzelnen Dienstanbieters die Anzahl der realisierbaren Anfragen stark einschränken. Ein beliebter Anbieter wird beispielsweise seine Batterieleistung schnell erschöpfen und dann für weitere Anfragen nicht länger verfügbar sein. Darüberhinaus beeinflusst die Skalierbarkeit eines Netzwerks die Dienstverfügbarkeit. Die Kommunikationspfade zwischen einem Dienstanbieter und Nutzer werden durch drahtlose Verbindungen, sogenannte "hops", realisiert. Lange Pfade mit zahlreichen hops sind anfällig für Störungen. Ein Dienstanbieter in einem MANET kann daher bereits als unerreichbar erscheinen, wenn lediglich der Pfad zu ihm zu lang wird.

Eine Erhöhung der Verfügbarkeit der angebotenen Dienste in einem MANET kann helfen eine insgesamt höhere Qualität der Applikationen im Netzwerk zu erreichen. Techniken der Replikation sind allgemein zur Verbesserung der Verfügbarkeit in verteilten Systemen besonders gut geeignet. Die meisten diesbezüglichen Ansätze basieren auf der Analyse und darauf aufbauend der Vorhersage des Zustandes des Netzwerks. Eine solche Analyse beruht üblicherweise auf Informationen, die aus den niedrigen Netzwerkschichten stammen. Die dafür nötigen Berechnungen sind allerdings aufwändig und erfordern oftmals die Nutzung spezieller Routing Protokolle oder spezielle Hardware wie beispielsweise GPS Module zur präzisen Ortung von Geräten. In diesen Fällen ist die Entwicklung von Ansätzen zur Replikation nicht nur sehr aufwändig, sondern zudem Architektur spezifisch.

Hier setzt die vorliegende Arbeit an. Sie stellt das Service Distribution Protocol (SDP) vor, ein Architektur unabhängiges Replikationsprotokoll für Dienste in MANETs das ausschließlich auf Informationen aus der Anwendungsschicht beruht. Aufbauend auf umfangreichen Evaluationen des Basisprotokolls werden verschiedene Erweiterungen vorgestellt und ihre Auswirkungen auf die Wirksamkeit des Protokolls untersucht. Dafür werden verschiedene allgemeingültige Definitionen im Umfeld der Dienstreplikation in MANETs eingeführt. Dies betrifft insbesondere das Konzept der Wichtigkeit eines Dienstes in einem Netz (service popularity) und verschiedene Maße hierfür. Darüber hinaus werden Messverfahren für die Optimalität der Verteilung von Diensten in einem Netzwerk vorgestellt. Bei allen Beiträgen der Arbeit finden die Auswirkungen der beschränkten Ressourcen der Protokollteilnehmer besondere Beachtung. Lösungen werden im Hinblick auf diese demonstriert und evaluiert. Im Rahmen einer weitergehenden Betrachtung wird das Protokoll zudem im Hinblick auf die Unterstützung der Ausführung von Workflows und Dienstkompositionen untersucht. Dies beinhaltet besondere Herausforderungen da nicht nur die Verfügbarkeit eines einzelnen Dienstes gewährleistet werden muss, sondern der einer ganzen Gruppe von Diensten, die zudem sehr heterogene Laufzeitbedingungen haben kann. Konkret umfasst die Arbeit die folgenden Kernbeiträge:

Das Design des vorgestellten Protokolls welches keine Analyse des Netzwerkzustandes benötigt. Die Integration des Protokolls in eine dienstorientierte Architektur wird diskutiert. Dabei benötigt das Protokoll lediglich Informationen über die Wichtigkeit einzelner Dienste. Diese Information kann aus der Anwendungsschicht gewonnen werden. Das vorgestellte Protokoll ist daher Architektur unabhängig und kommt ohne spezielle Anforderungen an die Netzwerkhardware aus. Die Wichtigkeit eines Dienstes wird über die Anfragecharakteristik der interessierten Dienstnutzer ermittelt. Zu diesem Zweck wird ein allgemeines Anfragemodell vorgestellt, welches es erlaubt diese Charakteristik zu erfassen. Verschiedene Szenarien für unterschiedliche Arten von Diensten (sehr wichtig, weniger wichtig, ...) werden vorgestellt und quantifiziert. Das Protokoll beruht auf drei Kernbausteinen. Der Replikationsmechanismus basiert auf dem Konzept der Dienstwichtigkeit. Wichtige Dienste werden mit höherer Wahrscheinlichkeit repliziert. Unwichtige Dienste werden über einen Ruhemodus deaktiviert. Falls möglich, werden deaktivierte Dienste mit Hilfe eines Caching Mechanismus zwischengespeichert. Durch das Zusammenspiel dieser Techniken bleiben wichtige Dienste im Netzwerk verfügbar und aktiv, während unwichtige deaktiviert werden um die beschränkten Ressourcen zu schonen. Die Leistung des Protokolls wird daher mittels der Dienstverfügbarkeit unter Berücksichtigung der verbrauchten Resourcen gemessen. Die Optimalität der erzeugten Repliken und ihrer Verteilung im Netzwerk wird ebenfalls untersucht. Für die Messung der Optimalität werden geeignete Methoden zur Erfassung der korrekten Zuordnung von Repliken vorgestellt.

Der Nutzen des Protokolls wird zunächst anhand eines einfachen Netzwerkmodells untersucht. Die Untersuchgung wird dann unter Nutzung eines detaillierteren Modells verfeinert. Die Vorteile eines dienstorientierten Replikationsprotokolls werden erläutert und diskutiert. Dabei werden auch umfangreiche Untersuchungen hinsichtlich einer optimalen Konfiguration des Protokolls durchgeführt. Diese führen zur Spezifikation mehrerer Erweiterungen. Dabei werden speziell die Auswirkungen verschiedener Modi bezüglich Replikation und Ruhezustand untersucht. Außerdem werden mehrere nutzenbasierte Koordinatorwahlverfahren und Dienstauswahlmechanismen hinsichtlich ihrer Effekte auf die Wirksamkeit des Protokolls evaluiert und diskutiert. Die Effektivität des Protokolls wird zudem mit verwandten Verfahren verglichen. Dabei werden auch Auswirkungen verschiedener Mobilitätsmodelle und anderer Kontextparameter berücksichtigt. Die Ergebnisse der Untersuchungen zeigen, wie das vorgestellte Protokoll eine optimale Verfügbarkeit der wichtigen Dienste in verschiedenen Szenarien erreicht.

Zur Evaluation des Protokolls werden verschiedene Szenarien für Umgebungen eingeschränkter Ressourcen entwickelt. Diese beruhen auf der Annahme, dass die Ressourcen der Dienstanbieter nur für eine beschränkte Zahl von Dienstanfragen ausreichtend sind. Jede Dienstanfrage verbraucht Ressourcen des Dienstanbieters und benötigt eine bestimmte Zeit zur Bearbeitung. In diesem Kontext ist die Behandlung von überlastszenarien essenziell. Daher wird ein nachfragegetriebener Mechanismus zur Lastverteilung vorgestellt, der diejenigen Anfragen verteilt, die die Kapazität eines aktiven Dienstanbieters übersteigen. Hierfür werden verschiedene Dienstauswahlmodi definiert. Ferner wird eine neue Definition für die Korrektheit der Allokation von Repliken vorgestellt. Die Evaluation der Methoden zeigt, dass die vorgestellte Lastverteilung eine hohe Verfügbarkeit und Allokationskorrektheit gewährleistet.

Neben den Einschränkungen bezüglich der Maximalzahl der möglichen Dienstanfragen an einen Anbieter werden ebenfalls Einschränkungen bezüglich der Zahl der auf einen Knoten replizierbaren Dienste berücksichtigt. Zu diesem Zweck wird ein Szenario vorgestellt, bei dem zwei Dienste im Konflikt um eine Ressource stehen. Es wird gezeigt, dass das vorgeschlagene Protokoll selbst in Szenarien mit einer geringen Zahl von Netzknoten eine hohe Dienstverfügbarkeit und Korrektheit gewährleistet.

Als abschließende Herausforderung an die Dienstverfügbarkeit wird die Ausführung zusammengesetzter Dienste untersucht. Zielstellung dabei ist die Gewährleistung einer hohen Verfügbarkeit nicht nur eines einzelnen Dienstes, sondern einer ganzen Gruppe von Diensten. Die Grenzen des vorgestellten Protokolls bei der Ausführung komplexer, zusammengesetzter Dienstanfragen werden diesbezüglich untersucht. Dafür wird die Qualität der Dienstausführung mittels verschiedener Güteanforderungen (QoS) in unterschiedlichen, zeitbeschränkten Szenarien bewertet. Außerdem wird ein Vergleich mit anderen Ansätzen zur Unterstützung zusammengesetzter Dienstanfragen vorgenommen.

Abstract

Mobile Ad hoc Networks (MANETs) are self configurable networks. A set of mobile nodes equipped with wireless communication capabilities can form a MANET. If two mobile nodes are located in the transmission range of each other, a wireless connection or link is formed. The set of the formed wireless links form the network topology. Since the mobile nodes are free to move, join and disjoin the network, the network topology is ever-changing. MANETs do not rely on any centralized administration or help to operate. Recently, MANETs have been gaining importance. The applications of MANETs are numerous and various. Personal area networks, Sensor networks and satellite networks are examples of this type of networks. The fact that MANETs do not rely on preexisting infrastructure may make them the only means of communications in some situations like search and rescue, disaster recovery and military scenarios.

The mobile nodes work in cooperative and collaborative manners to achieve the required core network functionalities like routing and data transportation. The available resources of a MANET like the offered bandwidth between nodes, the computation power of the mobile nodes, and the power resources like batteries are typically very limited and always varying. Sharing resources in this type of network is necessary. Service Oriented Architectures (SOAs) present a paradigm for achieving the required resource sharing in MANETs. If the resources to be shared are presented as service, the interested mobile nodes can access these resources through service requests. Recently, applying SOAs in MANETs has obtained a growing interest in research. SOAs offer many important features like the abstraction, encapsulation, de-coupling, discoverability, and autonomy which make them suitable for MANETs.

In general, the great challenge for deploying services in a MANET is the service availability. Service availability is negatively affected by the previously mentioned features of MANETs. A service provider may not be reachable by some or all of the interested clients for some time as the network becomes partitioned because of the mobility of the nodes. The service provider can have a lack of resources to serve more client requests. In MANETs, some service providers can loose their power rapidly because of the high client interest of their services. These exhausted service providers may be quickly not available for the interested clients. Moreover, the scalability of the network affects the service availability. The paths between a service provider and a client are formed of many wireless links or "hops". The paths with higher number of hops are more likely to fail. A service provider may appear unavailable to some clients because the paths to these clients are too long.

Increasing the availability of the deployed services in MANETs can ensure a better overall performance for different applications. Generally, in distributed systems, one way to achieve this increased availability is through replication of services. Most of the service replication approaches evaluate their replication decisions based on performing analysis and predictions for the information of the network status. This required analysis is usually based on extracting some information from the lower network layers. Performing these computations is exhaustive and requires presence of special components like routing protocols or General Positioning System (GPS) antennas. In these cases, the design of the replication approaches or protocols is not only complicated but also architecture dependent.

The main contribution of this thesis is therefore an architecture-independent service replication protocol for MANETs, Service Distribution Protocol (SDP), which relies on application-layer information, only. Throughout the thesis, this protocol and some enhancements are introduced and evaluated. Furthermore, this thesis introduces a set of formal definitions for the service replicability and replication process in MANETs. As an important contribution, the service popularity concepts and quantification methods are introduced. Also, measuring methods for the optimality of the service distribution in the network are proposed. The effects of the restricted resources from both client and provider sides on the proposed protocol are investigated and solutions are demonstrated and evaluated. Moreover, the proposed protocol has been involved to support the composite service execution process as a complex challenge for the availability of not only one service but also a set of services with different styles of execution flows. In more detail, the thesis consists of the following pivotal points:

- The design principles of the proposed protocol which enables to avoid performing any analysis for the network status. Integrating the proposed protocol to SOAs is discussed. The proposed protocol employs only information about the service popularity. The service popularity information is available at the application layer. The proposed protocol is architecture independent and does not require presence of any network components. The service popularity in this thesis is measured based on the requesting behavior of the interested clients. Therefore, a generalized requesting model is introduced to capture the requesting behavior of a client. A set of requesting scenarios for different

services (popular, less-popular,...) is proposed and quantified. The protocol has mainly three core mechanisms. The replication mechanism is based on the service popularity. Popular services are likely to be replicated. Unpopular services will be shutdown by the hibernation mechanism. If it is possible, the hibernated services may be cached by the caching mechanism. By performing sequences of these mechanisms, the service prevails the network with its replicas. Only the interesting (popular) services will remain active and others will be deactivated. Service availability and resource consumptions are measured in the performance investigations of the proposed protocol. The optimality of the generated replicas and how they are distributed in the network is also investigated. For measuring the optimality, a set of proposed methods for the correctness of the replica allocation process is contributed.

- The feasibility of the proposed protocol which has been investigated in a simple network model scale, then it is extended to a detailed extensive network mode scale. The advantages of being a SOA-based replication protocol are introduced and discussed. Investigating for the optimum set of specifications for the different parameters and mechanisms of the proposed protocol and their effects has leaded for introducing many concepts for the performance enhancements of the proposed protocol. The effects of different replication and hibernation specifications and behaviors are investigated. Moreover, impacts of a set of proposed interest-based leader election and service selection mechanisms (modes) on enhancing the proposed protocol performance are presented and evaluated. Performance comparisons between the proposed protocol and the other approaches are made. The investigations of the proposed protocol consider not only performance comparisons to the other alternatives but also applying different mobility models and settings on the performance. The results showed how the proposed protocol can achieve a relative optimum performance for the popular services in different operation scenarios.

- A set of scenarios for restricted resources which have been introduced to evaluate the performance of the proposed protocol. First, the resources at the service providers are asssumed to be enough for serving a limited number of requests. Each of the requests consumes some resources of the service provider and requires time to be responded. Managing the request congestion is a must in these cases. Therefore, an interest based load balancing mechanism is proposed to distribute the extra requests (over the capacity of the active service providers) to other providers based on the popularity and the proposed service selection modes. A new definition for the correctness of the replica allocation process is contributed. The performance evaluation results show high service availability and allocation correctness by the proposed

load balancing mechanism. Second, the availability of the required resources on the mobile nodes to participate in the replication process are restricted. A scenario of two resources' competing services is proposed. The proposed protocol keeps its high performance measurements for the service availability and the correctness even with low numbers of network participants.

- Finally, as a challenge for the service availability, the composite service execution is presented. The objective is to ensure higher availability for a set of services not only one service. The performance boundaries of the proposed protocol are investigated for the composite service requests. Quality of Service (QoS) based constraints for the composite service execution are investigated for the proposed protocol by applying a set of time constrained scenarios. Moreover, a performance comparison to other composite service execution approaches is made. A set of promising results for our proposed approaches is gained and discussed.

Contents

List of Tables

List of Figures

Part I.

Foundations

CHAPTER 1

Introduction

"The one thing we do not know
is the limit of the knowable"

(Jean Jacques Rousseau)

Cellular networks, Mobile Ad Hoc Networks (MANETs), wireless mesh networks, sensor networks, and satellite networks present examples of the variety of the wireless computer networks. They play a very important role in realizing today's communication. Small devices with wireless communication capabilities become pervasive in the everyday life. The technical abilities of these devices are increasing. These devices are various: Laptops, smart phones, Personal Digital Assistants (PDAs), small sensors, and even smaller personal devices like watches. These devices can form or be integrated in wireless computer networks. The ability of these devices to establish direct connections between each other represent the core of communications for MANETs. The applications of these network vary from battlefields and disaster recovery to connecting small room devices like computers, printers, ... etc. In these applications, the mobile devices cooperate to achieve the main network functionalities like routing, transporting data and sharing the distributed resources in the network. The distributed resources may be hardware or software. Data is not the only software resource considered, functionality sharing is equally important. One way to achieve this resource sharing is to present the different resources as services. Mobile devices can then request these services when needed. Service Orientation (SO) is a paradigm which employs a set of design concepts and standards for services and service management. These standards rule service designing, deploying, describing, discovery, and utilization processes. Service Oriented Architectures (SOAs) realize the service oriented computing in distributed systems. SOA technologies are realized for infrastructure-based networks like Internet. Recently, realizing SOA in MANETs has received increasing research attention. There adapting SOA components for MANETs is a must. Merging MANETs and SOA technologies is an interesting topic which enriches many fields like pervasive computing. SOAs enable decoupling the functionality and the platforms between clients and service providers supports the heterogeneous mobile architectures to share these functionality. Service orientation principles like discoverability and autonomy make SOAs a suitable candidate to rule the required functionality sharing in MANETs. The features of MANETs like the ever-changing topology, the limited and un-reliable resources like energy and wireless links, and the network partitioning behavior affect the service availability negatively and dramatically. Accessing the service at a specified server is dependent on the paths to this server. If these paths are changed, the service performance (like response time) will be changed as well. The servers may deactivate some service or disappear because of energy reasons. The service may be located in a different network partition from where the client requests are. This lack of service availability in MANETs has received an increasing research interest. Increasing the service availability in MANETs becomes an urgent need.

Resource replication in distributed system is one of the most classical topics in computer science. Replication can increase the service availability and make the system or the network more fault-tolerant regarding the errors of the lack of service availability. Service replication in MANETs is triggered to increase the service availability by interacting with the features of MANETs to cover and reduce their negative effects. Replication represents a good solution to realize reliable SOAs in MANETs. Service replication for increasing efficiently the service availability in MANETs under SOA standards is the main focus of this thesis.

1.1. Motivation

MANETs operate without any centralized administration or help. The mobile nodes should work in a cooperative and collaborative manner to achieve their required functionalities. Service replication can provide a technique for supporting service availability in MANETs. In order to increase service availability, existing replication approaches rely many analyses and processes for information about the network status like topology and links between the network participants to achieve the required replication decisions. Information about the network status is extracted from lower network layers. Extracting and dealing with this low level information is expensive in terms of time, required processing, and complexity. Most of the replication approaches which are based on network status analysis have the drawback of being network architecture dependent. While one of the main advantages of SOA is dissolving the coupling between the functionality (service) and its clients, a new source of coupling appears between the replication decisions and the network components. These approaches need to employ special network components like routing components to get information for their required analysis. MANETs they are a collection of heterogeneous mobile devices. Not all servers are relayed on the same network suite. The architectures of these devices are varying. One of our motivations in this thesis is to investigate the possibility to develop a network independent replication approach or protocol for service replication with SOAs.
Evaluating the performance of MANETs in general is an interesting research topic. Deploying the proposed algorithms, protocols and scenarios on real mobile devices becomes infeasible especially when the MANET size is not small. Computer simulation mostly provides the means for the required evaluations for MANETs. Designing experiments for evaluating and proving the concepts based on simulation tools is dominant in the community of MANETs. We are not only motivated to develop and use a suitable set of simulation tools to estimate and evaluate our concepts but also to survey, employ and compose measurements or metrics that can quantify the performance of service replication in MANETs. furthermore, we are motivated to compare the service replication approaches in MANETs (ours and the others) for the feasibility, the advantages and disadvantages, and the suitability for different scenarios from many perspectives.

Replication as a concept includes deploying the same resources several times in the network. Two extremes bound the replication process: with one replica, resources consumption is minimized, but so is service availability. With a replica on each node service availability is maximized, but so is resources' consumption. Awareness about resources have been employed from different perspectives like energy, memory and disk spaces by some of the service replication approaches. Considering the issues of the resource awareness presents one of our motivations. Finally, we are also motivated to investigate the effects and importance of increasing service availability on the composite service execution level. Composite service execution presents a challenge of a complex availability. By introducing the composite service requests, the availability of more than one service is required to be ensured. Considering increasing the service availability form the perspective of composite services represents a source of questions which have been addressed, discussed and answered by this thesis.

1.2. Research Goals

The previously mentioned research motivation can be translated into a set of research goals. The proposed research goals in this thesis are stated as follows:

Research Goal 1: To identify challenges, different approaches, and criteria of service replication in MANETs and to cover the related issues. The activities to be considered to meet this goal are:

- Defining the related issues of service replication in MANETs.
- Surveying the service replication approaches.

Research Goal 2: To develop a replication approach that can avoid the coupling between the replication decisions and the network status or components. The activities to be considered to meet this goal are:

- Investigating new sources which can deliver replication decisions.
- Developing replication approaches based on the proposed replication decisions.
- Evaluating the efficiency of the proposed approaches in different scenarios and establishing comparisons to other replication protocols.

Research Goal 3: To consider and investigate the resource awareness and composite service execution impacts of the proposed replication approach. The activities to be considered to meet this goal are:

- Developing of resource conservative scenarios.
- Investigating the effects of these scenarios on the proposed service replication approach.
- Introducing the problems of composite service execution in MANETs and the relation to service replication.
- Estimating the performance of the composite service execution with aid of the proposed service replication approach.

1.3. Contributions

The most important contributions of this thesis can be summarized as follows:

1. The Service Distribution Protocol (SDP) which is a service replication protocol to increase efficiently the service availability in MANETs. The proposed protocol can achieve high service availability and correct service distributions for MANETs. It does not require extracting any information from the lower network layers. Its replication decisions are based only on information about service popularity that can be gained at the application layer without performing any network status analysis. SDPs' core mechanisms, evaluation scenarios, performance comparisons to the other protocols, and awareness of resources are discussed in Chapter 4, 5, 7, 8, 9, 10 and 11.

2. Introducing formal definitions for the service replicability and the service replication process in MANETs. A set of common concepts which can distinguish the service and its replication process is introduced in Chapter 3. In Chapter 5, the replicable services and the service replication process are defined.

3. Introducing and quantifying the optimality of the generated service distribution by any replication approach as a set of performance measurements (replica allocation correctness ratios). These measurements are given in Chapters 5 and 7.

4. Introducing the service popularity as an important non-functional service attribute. Moreover, a quantification method based on the client requesting

behavior has been introduced. The proposed replication protocol (SDP) is based mainly on utilizing information about the service popularity. The importance of such a service attribute and how to capture it based on a proposed requesting model for the client requests is discussed in Chapter 6.

5. A general client requesting model with quantification features for the service popularity is introduced in Chapter 6. This requesting behavior model can capture the behavior of the request generation from a client to a specified service using a set of easy-to-understand parameters. This model can capture and model mostly the client requesting behavior to the most of the known services.

6. Employing the service popularity and the requesting behavior of the clients enable SDP to pay a variant replication effort for the different services. Not all of the deployed services will be replicated. Based only on their gained interest by the clients, services will be spreaded in the network. As mentioned in Chapter 6, SDP can enhance the service content (collection) in a network based on the concepts of popularity.

7. As previously mentioned, our work depends on simulation tools to evaluate of the proposed ideas and concepts. We introduced three simulation tools which are available online. The first tool is an event driven simulation tool which performs simulation for the proposed replication protocol on the level of the service requests. It assumes presence of suitable and efficient routing, transportation and scheduling network components. It simulates a real MANET for one SDP service application but in the best operational cases where the presence of packet loss is not considered. Nevertheless, the network participant mobility effects like topology changes and the network partitioning behavior are considered. Appendix A shows more details about this simulation tool implementation.
   The second simulation tool is implemented using the Opnet® Modeler® Wireless where the proposed replication approach is build as application module. This implementation can be used to simulate with different network components like routing components and physical layer probing approaches, and more other realistic scenarios. Moreover, this implementation enables modeling the resource consumptions by more than one simultaneously deployed service. This implementation includes also the load balancing mechanism which is preneted in Chapter 11. This simulation tool is published online as a contributed Opnet® model library[1] (for more details, see Appendix B). The third simulation tool is based on the second previously mentioned simulation

[1] `https://enterprise1.opnet.com/tsts/4dcgi/MODELS_FullDescription?ModelID=944`

tool but with a composite service execution considerations and capabilities. It is also published online as a contributed model Opnet® library[2] (for more details, see Appendix C).

8. We consider our publications as a measurement which can give an impression about the acceptance of our ideas, concepts and principles by the community. The publications out of this thesis are as follows:
   As book chapters: [HKR11] and [HKR09a].
   As conference full papers: [HDKR10], [HKR10a], [HKR10b] (under revision), [HKR10c], [HKR09d], [HKR08a], and [HKR08c].
   As workshop papers: [HKR09c], [HKRK07] and [HKR06].
   As technical reports and other publications: [HKR10d], [HKR10e],[HKR10f], [HKR09b], and [HKR08b].

1.4. Evaluation Methodology

In order to evaluate our concepts and ideas about service replication, we are motivated to use simulation as an evaluation methodology. By these simulation based evaluations, the performance measurements are observed against varying parameters. In the same simulation experiment, many measurements can be observed while varying only one performance parameter. In each of the evaluation experiments, which are shown during the next chapters, effects of one or more concepts of the proposed replication approach are evaluated and analyzed in terms of the performance metrics. We preferred to keep the required experiments and their results' analysis following to our proposed ideas and concepts and distributed in the different related chapters rather than to have one extensive evaluation part or chapter because the results and their analysis are too many to be inserted in one chapter. Moreover, it was necessary to have these distributed evaluations because some of the concepts, assumptions, and performance measurements are combined, enhanced or deprecated during the transition from one chapter to another. Finally, a common goals-results matching is given to reflect how the contributions meet the goal research in Chapter 13.

1.5. Structure of the Thesis

This thesis contains three parts. The contents of these three parts are as follows:

Part I is the foundation block of the work of this thesis. Besides this chapter, Chapter 2 introduces the set of related background topics of many areas for this

[2] `https://enterprise1.opnet.com/tsts/4dcgi/MODELS_FullDescription?ModelID=951`

thesis. Chapter 3 presents the literature review and the state of the art for the challenges and issues of service replication in MANETs. A special set of considerations which can describe the service replication processes in general is introduced and discussed. It presents a set of data and service replication approaches based on a proposed classification reflecting their different criteria and features. A comparison between some of these approaches is also highlighted.

Part II presents the SDP foundation in eight chapters [from Chapter 4 to Chapter 12]. Chapter 4 presents the basic ideas and abstractly the required core mechanisms to achieve the proposed replication principles. A simple network model has been used for the evaluations. The chapter ends with a set of requirements and assumptions to be considered from applying the pilot SDP implementations. Moreover, integrating SDP in one proposed SOA model is presented and estimated. Feasibility of the interactions between the proposed replication approach (SDP) and the SOA model components is presented and discussed. Chapter 5 gives the SDP core mechanisms in more detail and presents an extended realistic MANET model. It highlight the necessity of finding a good set of configurations for the client requesting behavior and SDP parameters. An initial consideration for the optimality of the generated service distributions is presented with experimental results. It presents a comparison in performance between SDP and other replication protocol from the related service replication literature [AYS+09]. The need of configuring the requesting behavior of the clients regarding a given service is highlighted. Chapter 6 introduces the importance of modeling the client requesting behavior, aggregates it into the service popularity, and shows the roles that the service popularity can play. Based on a proposed general client requesting model, two main scenarios of client requesting behaviors are proposed to be used in the next experiments. While Chapter 5 and 6 present the requirements, consideration and models for the client requesting behavior, Chapter 7 introduces the different effects of varying the replication and hibernation configuration on the performance of SDP. Moreover, it extends the optimality measurements of the generated service distributions and shows the differences impacts of four allocation correctness ratios. The leader election challenge in MANETs is introduced in Chapter 8 as a source for enhancing the SDP performance by introducing different hibernation modes. Combining the two proposed leader election modes showed better performance metrics and more correct service distributions. In Chapter 9, the questions related to applying SDP with different mobility models and the effects of the mobility on its performance are addressed. The effects of applying SDP with a heterogeneous mobility model are addressed in the first part of Chapter 9. SDP showed consistent performance regarding the proposed heterogeneous mobility and popularity settings. In the second part of Chapter 9, based on the comparisons of Chapter 3, two other service replication protocols are nominated [DB07, DB08a] to be compared to SDP. The comparisons showed how SDP can achieve relatively high performance in different scenarios of popularity and mobility. Chapter 10 presents other sources of enhancements for the SDP service selection process. The influences of generalizing the local availability of the cached replicas are presented as a selection mode. Other replica allocation concepts are introduced as a source for the increasing the correctness of the generated service distribution of SDP is proposed as a selection mode. The results showed how the first selection mode can enhance both service availability and distribution correctness, while neither the second mode alone nor combined to the first mode can enhance the performance of SDP. Moreover, the enhanced performance of SDP is compared again to the other protocols which have been used in the performance comparison of the second part of Chapter 9. SDP with the first proposed service selection mode is suggested to be used in the next experiments. The assumptions of the ability of the servers to serve all the generated requests in a network partition and the availability of the resources on the network participates which is made in Chapter 4 and Chapter 5 are criticized in Chapter 11. First, the ability of one replica to serve an infinite number of clients and their requests is shown as a function in the available resources by the servers. A problem of request congestion is assumed when a server can not serve an increasing number of the received requests. The capacity for serving requests can be fixed or dynamically specified based on the available resources by this server. Based on the service popularity and selection concepts, an interest-based load balancing mechanism can distribute the extra requests to other servers. The effects of having more than one active server inside a network partition on the correctness of the generated service distributions are covered and an allocation correctness ratio that considers the load balancing effects is presented. The results showed how this mechanism preserves a high performance for SDP with consideration of the request congestion problems. Second, the work of the restricted resources is extended by deploying two competitive services in the network. Once one of theses services is deployed on a network site, the other service can not be deployed there. Therefore, the network nodes with replicas of the first service can not participate in the second service replication process and vice versa. The experiments results quantify the negative effects of this service competition on the SDP performance. Moreover, the proposed load balancing mechanism is considered with the service competition and the results showed a good SDP performance. In Chapter 12, the composite service execution problem in MANETs is introduced with the related background. SDP can ensure higher atomic service availability versus features of the MANETs. As a challenge for SDP, how can it ensure availability for more than one atomic service regarding a composite service request. SDP performance has been tested versus different composite request lengths, QoS time-constrained composite requests and showed consistent and reasonable performance. Moreover, the SDP based composite service execution process is compared to other protocols and the results have been shown, analysis and discussed.

Part III In this part contains two chapters to summarize the contributions and the work of this thesis. Chapter 13 associates and collects the different results and contributions of this work through a common evaluation flow to validate the results to the initial research goals. Chapter 14 summarizes this thesis, highlight its results and contributions, and states the possible future work.

CHAPTER 2

Background

"Wisdom tree grows in the heart
and bears fruit in the tongue"

(An Arabic proverb)

In this chapter, the main topics relevant for the rest of this thesis are introduced. MANETs and their characteristics and applications are presented and discussed in Section 2.1. In Section 2.2, Service Orientation and the main processes of SOAs are demonstrated. Then, the challenges of applying SOAs in MANETs are presented. Replication for availability and replication in MANETs are discussed in Section 2.3. Finally, other related topics to this thesis are mentioned in Section 2.4. The work of this chapter has been partially published in [HKR06].

2.1. Mobile Ad Hoc Networks

Since 1970s, studies of different types of wireless networks in both military and civil applications have become increasingly popular in the network industry. Recently, wireless mobile networks have received more attention. By the beginning of 90's, early cellular mobile phones have been found independently in Japan and the United States. Wireless cellular mobile networks are designed to cover some predefined area, which is divided into smaller units with fixed stations called cells. In each cell there exists a fixed station that is responsible for organizing, controlling the communications, and supporting the mobile devices in its area. These fixed stations, known as "base stations", are connected to each other by more reliable connections like optical fibers.

Mobile Ad Hoc Networks (MANETs) represent an interesting category of mobile computing environments. In MANETs, the network participants have wireless capabilities and need to communicate without any centralized help or administration like in cellular mobile networks. In some applications such as battlefield communications and disaster recovery etc., only MANETs can provide the feasible means for communication and information access. This category of network is steadily growing in importance. From the scale of Bluetooth[1] [Mor02] devices at home, wireless sensor networks [SMZ07] and vehicular networks [OW09], to the scale of satellite networks MANETs are found everywhere. As shown in Figure 2.1, a wireless connections between two mobile nodes is established, if they are in the transmission range of each other. These connections are called wireless links. Wireless links are characterized by some metrics such as delay and bandwidth which are evaluated according to the environmental circumstances. As shown in Figure 2.2, A MANET could be considered as a cluster of mobile nodes or mobile hosts. The mobile hosts are free to move anywhere, join, or disjoin the network. The topology of a MANET is ever-changing and indeterministic. The established routes or paths in MANET between any two mobile nodes consist of many wireless links, which are considered as hops. So, MANETs are also called "multi-hop" wireless networks.

[1] `www.bluetooth.com/`

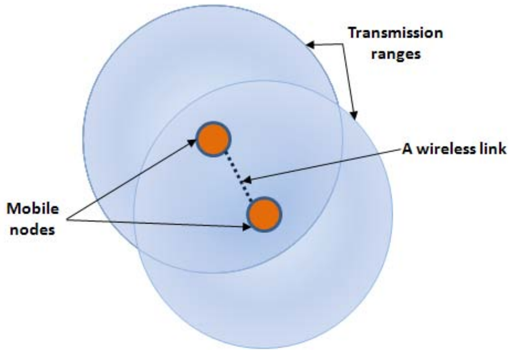


Figure 2.1.: A wireless link

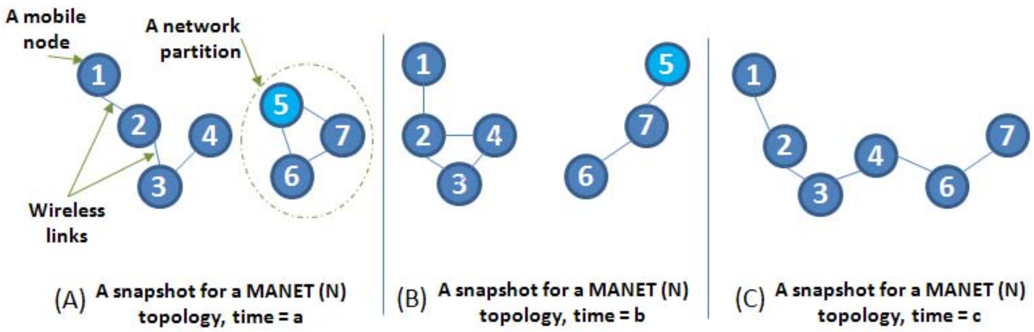


Figure 2.2.: A snapshots of a MANET topology

2.1.1. Applications and Technologies of MANETs

Applications: There are numerous applications for MANETs. Groups of moving soldiers in a battlefield communicate and coordinate with each other. A group of islands and ships communicates with the help of floating balloons and passing airplanes. Groups of people with portable computers share their data in a conference room. Students move in a campus with their laptops. Sensors (fixed or moving) in intelligent buildings can communicate using wireless connections. Cars in roads can communicate either to each other or GPS satellites. The common key factor of using MANETs is the fast deployment. MANTEs are infrastructure-less mobile networks that do not need complicated expensive instruments and devices to be established. This feature of MANETs gives this type of network increased attention. The presence of lightweight portable computers and other devices like smart phones with wireless communications capabilities represents the required basis for the practical mobile computing. Many businesses are dependent on distributed networked computing system over wireless communications such as multimedia interactions for mobile phones and their Web-based services. The underlying concepts of bandwidth, throughput, reliability, and cost obtain the main measures for the performance of wireless-based applications. The underlying MANETs technologies are quickly growing and facilitate more reliable new applications.

Technologies: The Linux Bluetooth-enabled watch (WatchPad®) of IBM®[2] which has been introduced through IBM® Pervasive, Mobile, Wearable Computing projects is a sample of the growing interest of realizing MANETs by research and industry. This is not new. It was accomplished until 2005. MANETs supportive technologies are growing and promising.
For the common applications of MANETs, Bluetooth and IEEE 802.11 [Gas05] are the best known and established standards. They set standards for the physical layer and medium access control (MAC) for wireless communications to realize MANETs. Bluetooth and IEEE 802.11 (commercially known as WiFi) are two communication protocol standards that define a physical layer and a MAC layer for wireless communications within a short range (from a few meters up to 100 m) with low power consumption (from less than 1 mW up to 100 mW) [FP05]. Bluetooth[3] has been realized by Ericsson® in 1994 as a standard communications protocol. It enables a set of short distance devices to be connected together. Based on low power consumption principles it can achieve short range communication for the low energy devices. On the other hand, IEEE 802.11[4] is a standard communication protocol for Wireless Local Area Networks (WLANs). It is the wireless application of the

[2] `http://www.research.ibm.com/WearableComputing`
[3] `http://www.bluetooth.com`
[4] `http://www.ieee.org`

IEEE 802 family of standards for Local Area Network (LANs). The most common versions of IEEE 802.11 are IEEE802.11b and IEEE802.11g. These two standards represent the most common wide spreading MANETs' standards of communications. Bluetooth connects close devices (not computers) and serving as a substitute for cables, while Wi-Fi realizes more the computer-to-computer connections as the WLANs require. Coexistence of both standards has been in many citations like [GVDS01, SSG07, LSN01]. [FP05] introduces a detailed comparison between these two standard families that highlights their main features and behaviors in terms of various metrics, including capacity, network topology, security, quality of service support, and power consumption.

2.1.2. Characteristics of MANETs

The dynamic characteristics of MANETs can be summarized as follows:

- Ever-changing topologies: mobile nodes are free to move arbitrarily. Thus, the network topology changes continuously. These changes may be taking place very rapidly. The wireless links are formed based on dynamic changing factors like the power of transmission and the link utilization. The topology changes includes the changes of the natures of the formed wireless links among nodes. The features of a wireless link can vary from one direction to the other one. Moreover, the mobility of the network participants causes the pretense of network partitions (see Figure 2.2). The network partitioning behavior is tightly coupled to the mobility patterns of the network participants. A MANET may have more than one network partition. These partitions are dynamic topology features. The formed partition may be merged or get partitioned depending on the formed wireless links between the mobile nodes.

- Link quality: wireless links are variable capacity channels. The capacity of wireless links is always lower than wired links especially in the multi-hop routes of MANETs. Link quality varies with high link failure rate due to many circumstances such as the radio-air propagation distance and power of transmission at host's antennas, as well as environmental circumstances like Sun activity, interference and presence of buildings.

- Energy constraints: mobile nodes in MANETs rely on batteries. Energy consumption becomes one of the main issues to be addressed.

- Limited resources: not only the energy is limited but also the other resources. The mobile nodes have less memory, disk space, and computational power resources. Moreover, the characteristics of wireless links make them show a poor set of link quality measurements like bandwidth compared to the other types of network connections [CS99].

- Limited security: MANETs are generally more prone to have physical security threats than fixed-cable networks. The eavesdropping [MP02] possibility is increased which should be carefully considered. Existing link security techniques are often applied within wireless networks to reduce the security threats.

In MANETs, in order to achieve the basic network functionalities such as routing [Ham04] and data transportation [HW01], mobile nodes should work in a collaborative manner in exchanging the network status information and forwarding the data (packets). However, collaboration is needed not only on the network layer's functionalities, but also on the application layer: the nodes should share their resources, i.e., the information and functionality they possess. Service orientation is a suitable candidate to be applied in MANETs in order to facilitate such resource sharing in these environments.

2.2. Service Orientation (SO)

SO is a paradigm for designing applications in distributed systems where the functionalities required in this system are represented as remote logic blocks (computing entities). These entities are represented by services. SO introduces the important background standards and principles to realize service-based applications. SOA is the underlying structure that supports deploying and managing services. SOA can be defined as "a framework for integrating and supporting IT infrastructure as standardized components (service) that can be reused and combined to address changing business demands" [BLJM08]. It can be also defined as a software architecture where the functionality is structured and realized as interoperable services [TK09]. SOA is not a concrete architecture but it leads to concrete architectures [Jos07]. The structure of any SOA application includes at least thee main parties as shown in Figure 2.3.

A client is supposed to be able to contact a centralized service repository to search and find its required service. The repository contains different offers for services which are provided by different service providers. Once the required service is found, the client gets information about the location of the service and how to interface its functionality. Afterwards, the service execution process between the client and the provider takes place.

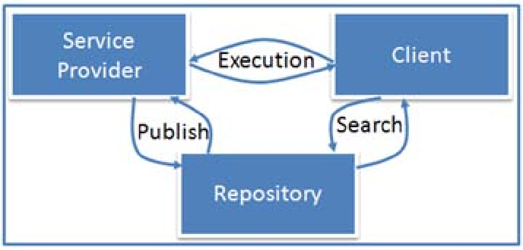


Figure 2.3.: A general SOA structure

2.2.1. Service Oriented Architecture (SOA) Principles

The services are assumed to be accessible through a computer network. As mentioned in [TK09], the principles of SOA can be summarized as follows:

- Loose coupling: dependencies between services should be minimized.
- Heterogeneity: SOA encompasses engineering a wide variety of services that have been deployed over different technologies like operating systems, frameworks, and languages.
- Abstraction and encapsulation: the services are implemented abstractly. The internal implementations of services are hidden. Moreover, the required information to allow service communication are required to be standardized. Each of the services should provide descriptions about their interfaces.
- Composability: the ability of assembling a set of services into complex composite services to provide some complex functionalities.
- Autonomy: the delivered logic should be controlled by the services.
- Discoverability: based on the available service descriptions, services can be found. As mentioned before, the service repository (which contains the service offers or descriptions) can be searched for the required functionality.
- Platform independence: the services are hosted by different providers. Each of theses providers may differ in its architecture, operating system, the developing language of the services and their interfaces. SOA applications do not require any of the clients and providers to be aware about these issues. They can communicate using open standards and protocols and common data structures and formats.

The previously mentioned principles enable any SOA to combine a set of standards and protocols that can be used to support, manage, coordinate and enhance achieving the required functionalities of SOA-based applications.

2.2.2. Main Processes of SOAs

- Service Description: accessing data and functionality is obtained by posing queries (service requests) to be matched with service offers. The more precise queries and offers are described, the better matches can be achieved. Semantics are required to enrich the service offers to be easily and correctly matched. Service description languages [New02] provide different paradigms for describing the service functional and non-functional properties [HKRK07] and their semantics. A good service description language is therefore the most important basis for service-oriented architectures [HKR06]. Not only the services offers are needed to be modeled based on some ontological standard of the service description languages but also the client requests.

- Service Publishing: the service providers are supposed to upload the service descriptions or offers to be stored in the service repositories. The offers in the service repositories are searched to be matched with the client requests.

- Service Matching and Discovering: matching the client requests against the stored service offers at the service repositories is the core of the service discovery process. This process supplies the information about the required interface [New02] for the client to start the service execution process.

- Service Execution: the service execution starts once the client sends the service request to the service provider. This process includes interpreting, serializing the request parameters, allocating the required resources, triggering the service executables and finally sending the results to the client.

- Service Composition: sometimes, the required functionality by a client can not be provided by one single service provider. In these cases, a set or some union of service providers should deliver such a complex functionality. Therefor, clients compose composite (complex) service requests. The composite request is posed to many atomic services (accessed on a specified provider). The composite request should be complemented with execution plans that provide information about how to control the flow of the requests to theses atomic services (execution process).

- Application centric: Not only the users (human client) communicate with the services but also the services can communicate with each other. This communication is done automatically upon the required functionality for some SOA application.

2.2.3. SOA in MANETs

In MANETs, the mobile nodes have to work in cooperation to achieve the required networking functions such as routing. Sharing resources is vital for this kind of collaborative network. The available resources on the mobile nodes of the network are needed to be utilized in a cooperative and collaborative way to preserve the network main functionalities. What is meant by resources is not only the computing power, memory buffers, and bandwidth of the links, but also data and functionality. The integrity through sharing the resources between the (low-resourced) mobile nodes is necessary. SOA provides common principles for sharing resources in MANETs as services, if whatever resource is presented as a service. If the mobile nodes agree on advertising their offered functionality and other resources as services and to include functionality offered by others as service requests, resource sharing becomes possible. Moreover, if services are described semantically, this resource sharing can happen dynamically without the need for human intervention.

2.2.4. Challenges

[Han06, CC05, JDT05, KT03, Han05, Jus05] address challenges of deploying SOAs in MANETs. As the service represents the core of SOA, it is required to be available and accessible under the previously mentioned SOA principles. The service availability represents the main challenge for SOAs to be addressed in MANETs. Not only the network partitioning behavior makes some of the deployed services in the network unavailable for some time for some mobile nodes, but also the limited resources that the mobile service providers rely on. These limited resources make the deployed services unable to serve all of their clients even if they are in the same network partition. Moreover, special considerations are needed for the process of service offers' publishing. As mentioned in Section 2.2, centralized service repositories in MANETs are not recommended. Due to the decentralized nature and the high possibility of single node failures, centralized repositories which are assumed to be deployed at one node are not suitable. The challenges of realizing SOAs in MANETs can be summarized as achieving *Availability*. The availability has many dimensions. While service availability affects the processes of service execution, the repository availability affects the service publishing and discovery processes.

2.2.5. Web Services

Web services is a term that combines the service to its access method. A Web service is a service implemented to be accessed on the Internet. The Web service uses a set of Extensible Markup Language (XML) based standards to realize the concepts of SOA. Accessing remote functionality in the Internet became feasible by the presence of Web 2.0. Electronic Data Interchange (EDI), Common Object Request Broker

Architecture (CORBA), Distributed Component Object Model (DCOM), Remote Procedure Call (RPC) , and Java Remote Method Invocation (RMI) are standards enabling accessing remote functionalities and resources over distributed systems, networks and the Internet [Fon04]. These technologies still exist today but failed to gain significant market share due to cost,complexity, flexibility, industry support, and compatibility issues. The main advantage for Web services technologies over these technologies comes out of the fact that any system capable of communicating via a standard Internet transport protocol can communicate with a Web service [Cha04].

2.3. Replication in MANETs

In the areas of distributed systems and database, replication has classically been introduced as an approach to enhance the performance by increasing the resources' availability and accessibility. Replication means realizing a set of redundant resources within the same system. Replication can be used to improve service availability under both system and network failure conditions. It can also be used to improve the scalability of services that have a large number of users [Cha05].
Replication relies on different assumptions and offer different guarantees for the clients [WPS+00]. Not only availability and accessibility issues are targeted to be enhanced by replication approach, but also other issues like fault tolerance [WPS+00, GB99]. In our research we will apply replication for increasing service availability and accessibility for clients in a MANET. [WPS+00] introduces a definition for the replication as an abstract problem in distributed systems. It provides a set of characteristics that can categorize the features of the replication process features in distributed systems and databases.

2.3.1. Replication for Availability

If there is a replication approach that can produce replicas of a specified entity (service or data) and distribute these replicas in different locations of the network, data items or services can remain available through their replicas for clients when part of the network looses the access to these original data items or services. Replication for availability is an especially important principle in MANETs, where the failures and loss of access happen frequently. Many factors can affect increasing the unavailability of remote resources like services in a MANET. Running out of energy, the length of paths to resources, and network partitioning behavior have dramatical effects on the availability in MANETs. [PGVA08, DB09] introduce a survey that classify data and database replication protocols mainly based on these previously mentioned factors.

While replication is a classical approach for increasing availability in many environments, unfortunately, none of the well known replication methods from systems like Data Base Management Systems (DBMS), Redundant Array Of Inexpensive Disks (RAID), and Domain Name Server (DNS) work under the constraints given in MANETs. However, the problem of service replication receives potential research effort. As we are going to show in Chapter 3, there are approaches that address the issues of replication for service availability in MANETs like [HKR08c, AYS+09, HSC01, DB08b, DB08a, WL02, JJKY04, DJ07, DB07, JW05, DJ07].

2.3.2. Performance Evaluation for Replication

Imagine the best case from the availability perspective, that is to have replicas of all the required functionality and resources distributed on each network participant's side. Of course, this not only presents waste of resources but also requires a huge effort to keep these replicas consistent and synchronized. It is clear that there are some constraints needed to be drawn to keep any proposed replication mechanisms. Increasing the availability should be achieved with the minimum replication cost. The correctness of the replica allocation process should be high as much as possible. The number of the completed requests for the service and its replica should be indicated. Evaluating the performance of the applied replication algorithms, approaches, mechanisms and protocols presents an interesting topic to be covered in the rest of this thesis.

2.4. Other Topics

In this section, other related topics in evaluation, special problems of services and MANETs which are relevant for our research are presented.

2.4.1. Evaluating Performance of MANETs

Starting form teh mid 1990s, MANETs evaluation presents a hot topic which receives increasing research effort. Performance of the proposed MAC and physical layer models, routing, data transportation, and scheduling protocols in MANETs introduced a material for introducing the required evaluation techniques of MANETs. Evaluation is very important to reach judgments about whatever proposed solutions from different perspectives. Mathematical and statistical models for the network components can provide some analytical guidance [DPM02] for some evaluations. Moreover, capturing some measurements about a network then doing analysis for this data from specified perspectives is one of the most and direct evaluation methods. It requires only a precise definition for the required measurements. Sometimes, especially when the scalability dimension is included, capturing and doing analysis for some measurements id neither sufficient nor meaningful. Therefore, in these cases, other approaches are required for evaluation. Simulation [McH91] can cover beyond the abilities of the other approaches. Traditional scientific knowledge has generally taken the form of either theory or experimental data. However, where theory and experiments stumble, simulation may offer a third way [K0̈7]. Simulation is the use of a model to develop conclusions that provide insight on the behavior of any real world elements. Computer simulation uses the same concept but requires that the model be created through programming on computers [McH91]. Providing computer simulations and tracing their results is one of the basic milestones in evaluating the proposed approaches of MANETs. For that purpose, a set of distinguished network simulators have been introduced. NS2[5], Opnet®[6], QualNet®[7], and GloMoSim[8] represent the most commonly used simulators by both researchers and manufacturer of MANETs. [HBG06, KCC05] present some features and statistics about the publications and contributions which use a set of different network simulators in MANETs. In this thesis, we will use the computer simulations as an approach of evaluation our proposed solutions.

2.4.2. Leader Election Problem

Leader election is a fundamental problem in distributed systems. In MANETs, where failures are considered the norm and not the exception, leader election is very important issue to be considered [DB08b]. Leaders are required in most of the distributed systems as the developed algorithms are mostly not symmetric with respect to the responsibilities to be achieved by the system participants. Many applications of MANETs such as key distribution [MP02] , routing coordination [SSB99] , sensor coordination[SMZ07] and group communication service [DB08b] require the presence of leader election mechanisms. Regarding service replication in MANETs, the leader election problem has been introduced [DB08b, DJ07]. From many perspectives like the available resources [DJ07], topological and communicational [DB07, JW05], mobility [HC06], and service popularity [HKR08c] issues, a set of leader election algorithms have been introduced with replication in MANETs. In Chapter 8 more consideration will be given to the leader election in our research.

[5] http://www.isi.edu/nsnam/ns/
[6] http://www.opnet.com
[7] http://www.scalable-networks.com/
[8] http://pcl.cs.ucla.edu/projects/glomosim/

2.4.3. Load Balancing

Since the service requests (load) at some of the service providers may be higher than at others, the load at some providers may exceed their abilities to serve. The excess of the load at some service providers presents a congestion. A congestion affects negatively the performance of the service. It also leads to request drops and increased delays, and tends to be grossly unfair toward some clients. Load balancing between different service providers is required to enhance the overall service performance, especially when replication is considered. Service replication produces a set of replicas. If it possible to distribute the request load over several replicas in a way that all of these requests are served efficiently and considering the resource utilization, the overall service performance may be enhanced. Load balancing as a solution for the congestion problems has been applied by many approaches in MANETs. More details about our research's consideration of request congestion issues and proposed load balancing-based solutions are addressed in Chapter 11.

2.4.4. Quality of Service (QoS) and Resource Awareness

The applications in MANETs require guaranteed boundaries of acceptable quality and performance. The notation Quality of Service (QoS) has been proposed to capture the qualitatively and quantitatively defined performance characteristics of a given service [Che99]. As defined in [GJ79], QoS defines the nonfunctional characteristics which affect the perceived quality of a given system output. The QoS requirements can be translated into a set of given constraints. These requirements are realized in resource reservation schemes and time constraints to ensure the required quality. The main factors that provide specifications for these requirements are a set of parameters such as bandwidth and delay. In our research, QoS has played a role in the form of constraints on delays for the response time of composite service requests. In Chapter 12, we investigate the impacts of restricting a set of composite service requests to be executed within certain time constraints.

In MANETs, the limited resources and their availability require efficient utilization. Avoiding the deploying resource exhausting approaches is a must. Therefore, in Chapter 11 issues of the resource restrictions are addressed. A set of dynamic resource changing scenarios have been introduced to examine a set of proposed solutions in more realistic situations of MANETs from the perspective of resource awareness.

2.4.5. Composite Service Execution

Service composition is one of the main processes involved in any SOA. Some requests are composed of different required functionalities. Each of these functionalities is hosted by a service provider. The service providers are not located in the same mobile node. Therefore, the composite requests which contains a set of simpler request services may be generated at some client. Actually, these composite requests include also specific information that controls the flow of the simple contained requests to the different services. Controlling the flow of the simpler requests in a composite service request is done during the execution process. In composite service execution, more than one service is required to be available, otherwise the composite service request fails. Therefore, composite service execution represents a hard challenge for any service replication approach. In Chapter 12, the background and issues related to the composite service execution are addressed and investigated with our proposed service replication proposed solutions.

2.5. Summary

In this chapter, the related background and topics have been introduced. Wireless mobile network and MANETs have been presented. MANETs with their applications, common technologies, and characteristics have been mentioned. The role of services in MANETs has been discussed. Service Orientation (SO) and Service Oriented Architectures (SOA) with their main principles and processes have been discussed as candidates for managing services in MANETs. Applicability, suitability issues and challenges of realizing SOA in MANETs have been introduced. One of the major challenges in these network is the unguaranteed and low service availability. Service replication has been introduced as a technique to increase and ensure the service availability. Issues related to service replication and its evaluation have been presented. Finally, other related topics of network management such as leader election, load balancing, Quality of Service (QoS), and SOA service availability for composite service execution have been highlighted with respect to the flow of the work in this thesis. More details about these topics will be given, when required, in the context of the next chapter.

CHAPTER 3

State of the Art

"Four things can not be hidden for a long time: the science , the stupidity , the wealth and poverty"

(Averroes)

The previous chapter introduced the necessary technical background of this thesis. This chapter presents the state of the art in service replication and its related work.
Replication is one of the classic computer science techniques which has been used in many areas like databases and distributed systems. The purposes behind applying replication are various. Replication can fit mainly where fault tolerance, backup, reliability, availability and other performance purposes need to be enhanced. Replication in computer networks has been employed early by many applications. Replicating data as resources is one of the oldest challenges in distributed systems. Not only data as resources here to be replicated in a distributed system but also the offered functionalities. Functionality replication can ensure a better functional availability and fault tolerance of the system. Services combine both of data and functionalities into one structure. Service replication can be understood (as we are going to show) as a subset of data replication. The service replication issues can be seen as data replication with some further requirements and features. In MANETs, data replication issues are influenced by the nature of these networks. The impact of MANETs' nature like mobility, limited resources and ever-changing topologies are the main factors that represent a set of constraints and requirements to be considered by any data replication approach.

The work of this chapter has been partially published in [HDKR10] and achieved in a cooperation with Dr. Abdelouahid Derhab from Department of Computer Engineering, CERIST Center of Research, Algiers, Algeria. This chapter is organized as follows: In Section 3.1, the used terminology are presented. Section 3.2 presents and introduces the importance of data and service replication in MANETs. The issues that affect data replications in MANETs are presented and discussed in Section 3.3. In Section 3.4, a proposed classification for the data replication approaches in MANETs is presented with examples. Section 3.5 considers the issues related to the service in the replication process. A set of metrics that are used to quantify the replication performance of the different approaches are presented and discussed in Section 3.6. The related areas to our work are presented in Section 3.7. Finally, the work of this chapter is summarized in Section 3.8.

3.1. Terminology

In this section, we present the frequent used terms in the rest of the thesis.

- "*Replica*" is a term which indicates an identical copy of a data item or service hosted or to be hosted on a mobile node which is not the original node that hosts the original data item or service. The original data item or service represents the initial source of this replica.

- "*Server*" is a mobile node that hosts a replica actively and allows the other nodes to access this replica by posing their requests. A server can be also called a service provider.
- "*Client*" is a mobile node which generates requests to be posed to a server. Regarding the same service, a mobile node can act as a server and client at the same time.
- The "*replica allocation or placement*" is the process of assigning a generated replica to a mobile node which turns into a server.
- The "*Replica deallocation*" is the process of deactivating an active server. It may include deleting or caching the active replicas.
- A "*service or replica distribution*" is the situation of the replica allocation in a given network. The service or replica distribution can be defined as the set of the generated replicas of a given data item or service relative to their current location and allocation status at certain time.
- A "*connected session*" is a virtual connection between a client and a server which may have different statuses and attributes based on the client request parameters regarding a specified service.

3.2. Data Replication in MANETs

As mentioned in Chapter 2, replication is one of the techniques that can be used generally for increasing data and service availability in both wired and wireless networks. The challenging features of MANETs affect deeply on the data and service availability. The frequent failed wireless links, the partitioning behavior and the limited resources decrease dramatically data and service availability in MANETs.
Data and service availability can be explained from many perspectives. Data replication represents a superset from a replication perspective. Distinguishing considerations between data and service replication are addressed in this section. [PGVA08, DB09] include a set of issues to be addressed in data replication in MANETs. [MPV06] introduces a survey on Peer-to-Peer (P2P) replication systems and highlights the features and criteria for management of the replication processes. MANET and P2P systems exhibit many common properties that justify describing them in one article. Common properties include a dynamic network environment with high levels of node churn, changing network link quality, and variation in the amount of resources nodes contribute to the system. These properties have led to the development of techniques that allow decentralized coordination of entities and their continuous adaptation to a changing network environment [BDS07]. The

issues concerning data replication of [PGVA08, DB09, MPV06] can provide a general classification for the data replication approaches in MANETs. We complement these criteria by the service unique concepts. Moreover, we merge and conclude these criteria in the following subsections.

3.2.1. Mobility and Partitioning Behavior Issues

In MANETs, the mobility of the network participants affects negatively the data availability. Paths or routes with higher number of hops to access far data items are likely to be broken. Not only the length of paths but also the dynamic status changing intermediate nodes which participate in these long paths affect negatively on the data availability. Moreover, Due to the frequent behavior of forming network partitions in MANETs, some servers are not reachable from some mobile nodes in other network partitions. Deploying replicas in the ongoing to be formed network partitions increases the data availability. Some data replication approaches try to predict the network partitioning behavior and deploying replicas inside the ongoing network partitions before the partitioning behavior completes.

3.2.2. Energy Issues

Since both of mobile servers and clients are relaying on limited energy resources like batteries, applications of MANETs should be energy-consumption-wise. Servers may be exhausted due to serving many clients. The servers about to be exhausted may affect the data availability. Balancing accessing the different servers in a network may improve the overall energy consumptions by the servers. Some replication approaches can distinguish that some servers are about to be exhausted and try to provide replicas on new nodes in order to ensure their availability. Some of these approaches take care about the produced replica distribution ensure lower overall energy consumptions.
From a client perspective, allocating nearer replicas may ensure better response times. The clients may loose their power waiting for results of some queries to servers. Some replication approaches consider allocating replicas nearer to their frequent clients to save their energy.

3.2.3. Scalability Issues

The number of the mobile nodes in a given MANET (network size) play an important role as a trigger for the replication. As the network size increases the number of the clients to access some servers inside the network increases too. The ability of these servers to serve this increasing number of queries becomes the main question. As the number of the queries regarding a specific server is increased, its ability to serve new queries decreases. At some time, if this increase continues, the server will no longer available for new clients. Moreover, the presence of long paths with high number of intermediate nodes in such dense network sizes increases the path failure possibilities. Moreover, the slow and inefficient response of the client queries will increase. Some of the replication approaches address the issues related to the scalability to be the main issue of their data replication decisions.

3.2.4. Managing Replicas

The following points are important managing issues to be considered by the replication approaches in MANETs.

- Optimistic or Non-Optimistic: Optimistic replication approaches expect that conflicts between concurrently running replicas will be rare and have no effects on the replication consistency. The non-optimistic approaches expect the opposite and introduce conflict-resolving mechanisms to keep the consistency. From a database perspective and for data consistancy, client queries to a server can be reduced to be either read, write, commit, or abort transaction. Transaction as a term comes from the field of database and concurrency control [EN06]. It indicates the whole required operations to be done to meet a query at the server. Consistency in some applications are vital. How to manage the concurrent replicas and keep them consistent is an important issue. Regarding the partitioning behavior of MANETs, two network partitions may remain separated forever each with its own replica. Each of these replicas operates under different sequences of transactions. In this case, while replicas are in separate partitions, no recovery mechanism can be applied to keep consistency. Therefore, data consistency cann't assumed to be 100% in MANETs. How are update messages exchanged between the different replicas? This question is very important to be answered by any data replication approach for in MANETs.

- Full replication or partial replication: While full replication denotes that the data are to be replicated over all available nodes in the network, partial replication refers that the replicas will be hosted on some of the mobile nodes.

- Synchronous vs. asynchronous: a synchronous replication approaches assume a direct and correct updating for all replicas per each replica update. On the other hand, an asynchronous replication approaches does not require (some of/all) replicas to be updated at least for sometime.

- Pull-based or push-based: In pull-based replication approaches, clients are supposed to ask for hosting their own replicas from a server. In contrast, push-based replication approaches let servers dominate the whole replication process and the allocation of the new replicas.

3.2.5. Service Awareness

In addition to the previously mentioned criteria, we add an important criterion to highlight the ability of a specified approach to deal with services during the replication process. In this subsection, the special considerations for service replication are addressed and discussed. In general, the features that influence a special consideration for the service replication awareness are as follows:

Entity and structural: In data replication, data items are the targets to be replicated for the replication process. On the other hand, replicating services (e.g. executable files) to new sites does not ensure producing the desired running replicas. The new allocated replicas not only require executable files but also other resources. These required resources may be offered by their new location, replicated to their new location or accessed remotely on other network parties.

Operational: Two types of operations performed on the data items. These operations are the *query* and *update* operations [DB09]. In query operations, a client will be informed about the value of a data item. In the update operation, a client enforces a data item to change its value. On the other hand, accessing a service by a client is done by posing *requests*. A service as an entity consists of a set of data items, resources (local or remote), and connected sessions to the clients. A service can be considered as a complex data item which requires special awareness during the replication process.

Functional: A service request may have different input arguments. The service requests may trigger different flows of processes or functionalities at the service provider sides. The resources regarding each service request may differ even if the requests are similar. For example, consider a provider offering a map download service and the deliverable is either free maps (where not many details are described on the map) or maps with some charge. A similar map request (for this provider) with similar process flow allocates different resources. Regarding the connected sessions to the services, some services require that some information about the service state should be available for the clients and parameterizing the connected session. The service responses depend not only on the input arguments of the requests but also on the current session state. [TMMF07] introduces the importance of considering and managing the *stateful* service in reality. The *European Network of Excellence in Software Services and Systems* project defines the stateful services as follows:

> It is a service that has a current state. The execution of one of its operations during the time returns different outputs even if invoked with

the same inputs. Moreover, some relations can exist between the service operations requiring them to be invoked in some specific order[1].

Some service are considered to be stateful services [MHKS08]. In [TMMF07], some payment Web services are considered as mandatory security authentication services. These services label the user sessions with different state attributes. The services treat the same requests (from different sessions or clients) differently based on their sessions' states. [MHKS08] shows how important it is to deal with the data of the application (service) state as a dynamic data to be replicated and introduces the related issue of the required replication process for such states.

Replicability It is trivial that data items are always replicable. Are services always replicable? Most of the service can be replicated theoretically. In reality, the feasibility becomes the judgment. For example, theoretically it is possible to replicate a print out service by installing a new printer somewhere in the network but is it feasible can this be achieved automatically?. Another example, in a MANET, at sometime, some participants have some specific hardware capabilities like GPS antennas. One of these participants offers a "Geo" service that enables the other participants to determine their relative geo-locations . This offered service will remain replicable only if there is at least one of these GPS-antenna equipped participants reachable and joined in the network by the beginning of the replication process. Another example, some remote resources are allowed to be accessed by a service provider. If one of the other network participants is not allowed to access these remote resources, this service replication process at this participant may produce incomplete and incorrect replicas. Therefore, it is not always feasible to achieve the process of service replication. A replicabale service in some scenarios and context becomes un-replicable in other scenarios and context. More details about service replicability are introduced with a formal model in Chapter 5.

Service Model Figure 3.1 presents a service model which contains data items, functions, and flow specifications. The service is hosted and utilizes some resources of the hosting node. The service may access other remote resources. The service has many functional and non functional attributes like cost. The clients are connected to the service though "sessions". As mentioned before, these sessions hold the state attributes. Based on the available resources of a provider, once a service request arrives, a connected session is established. The session's function is to hold the required information about the allocated resources for serving the request until the service response is formed and sent. More consideration about the connected session are introduced and discussed in Chapter 11. If required, The quality of the

[1] `http://www.s-cube-network.eu/km/terms/s/stateful-service`

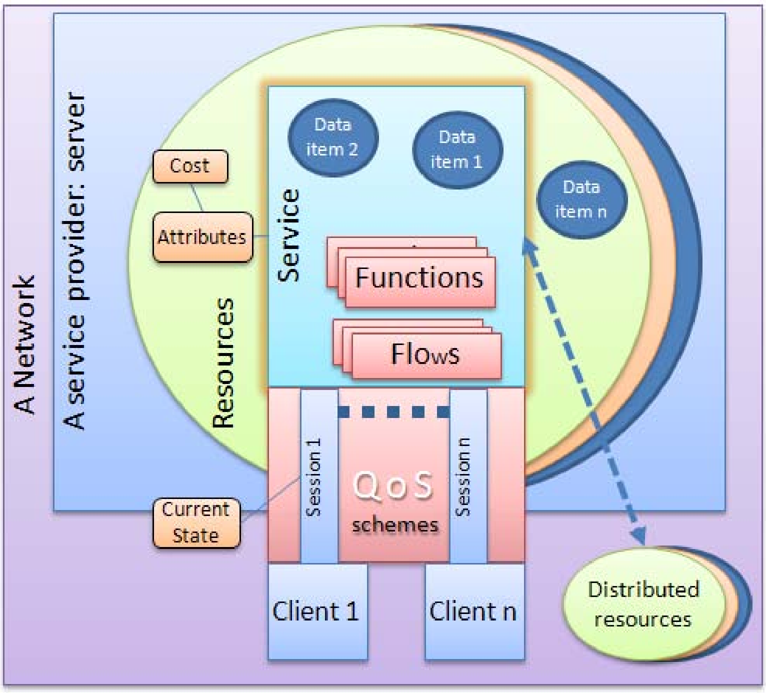


Figure 3.1.: A service model

service execution process is monitored and controlled by the provided QoS schemes.

Service Replication: Impacts on Replication as a Process Service replication in can abstractly be mapped into data replication. As previously mentioned, we are convinced that the service in principle can be represented as a complex data item. Each of its components needs special treatments during the replication process. The difference comes from the fact that service replication includes different dynamic data sets to be replicated. The service executables, service states, the required data resources such as files, information for resource allocations by the new replica location, specifications like the privileges and permissions for the remote resources, and the dynamic data of the connected sessions which include the service execution's current states. The data sets (regarding the same service) to be replicated have different specifications like the size in bytes and different structures, constraints and privileges (permissions) during each replication process. A service replication approach in MANETs should consider not only the different and dynamic data sets required to be replicated but also the feasibility of replication at the proposed replica's new location.

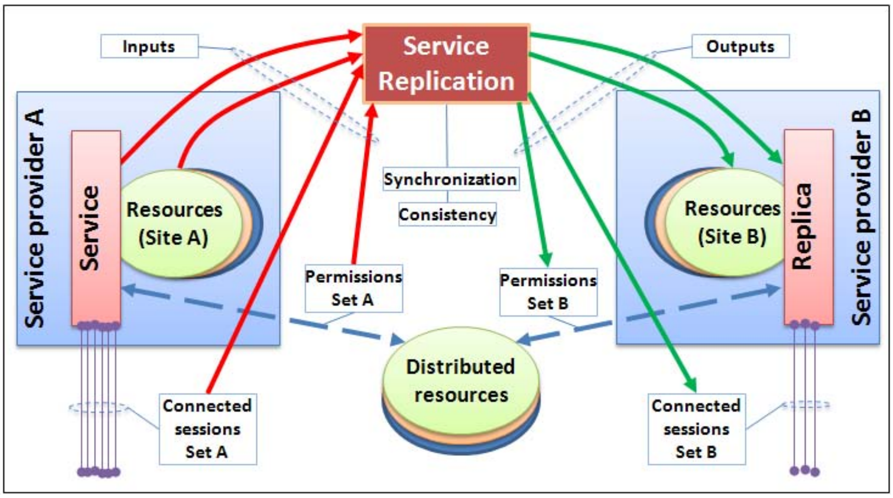


Figure 3.2.: The service replication process: Inputs/outputs

Service Replication Process Figure 3.2 shows the inputs and outputs of a service replication process. Based on this figure, a definition for the service replication process can be introduced. It can be defined as the replication process that can replicate and allocate a service, as defined by our service model, from a site to a remote site. The replication process should not only replicate the required data items, files and executables but also the required (or the minimum subset of) permissions and connected sessions to keep the new-generated replica running with acceptable level of correctness. The client sessions may be migrated transparently between service provider like in [Han05]. The replication process should show a policy for calming the service synchronization and consistency. More details about the service replication process are introduced with a formal model in Chapter 5.

Impact of Service Orientation: Data to be Services As mentioned in Chapter 2 the available resources in a MANET can be represented as services. Since accessing remote data is one of the most important requirements by many applications, these remote data can be represented as services. On the other hand, not all services can be presented as data.

3.3. Replication Design Requirements

In this section, a list of requirements of the replication process which includes the main questions to be posed when one tries to design or select a replication based approach for data or service availability in MANETs is covered. These requirements can describe another set of criteria that can describe the service process. The set of questions is as follows : Under which events will a new replica be created? Where will a new replica be placed? How are concurrent replicas being managed? What type of replication is being used? During the next subsections, more details about these considerations are given.

3.3.1. Replication Decision

As mentioned previously, the issues behind the replication decisions have been covered during subsections 3.2.1-3.2.5. In this subsection, we normalize these issues to have two dimensions: time and location. The term "replicator component" refers to the component that is responsible for the replication process. This component can be centralized at one mobile node or distributed over more than one node to evaluate the replication decisions.

Time

Producing replicas is controlled by a set of events that reflects the current status of the environment. At the occurrence of some events (individually, mutually or concurrently) the replicator component decides to produce a (set of) replica(s) to maintain higher service availability. In MANETs, the partitioning behavior is a popular candidate to be used as an indicator for the necessity of the replication process. Based on the prediction of a newly forming partition, the replicator component tries to push a replica inside it before the partitioning process finishes. Other approaches, as in [HSC01], especially for dense MANETs, try to keep the number of hops to reach the service as small as possible. Therefore, the replicator components spread replicas to the far parts of the network to keep the quality for accessing their services higher. Also, service popularity has been introduced as an indicator to be taken into consideration during the replication process. Other approaches like in [HLN03] evaluate the frequency of access to decide when to replicate.

Location

The replica allocation (placement) process complements the replication process. It allocates the new replica to a place in the network. Partitioning behavior of MANETs, avoiding long paths to a service, and saving the energy of both clients and providers affect the replica placement process. Regarding the natures of MANETs, the efficient replication process of services in such networks requires continuous replication decisions to ensure higher service availability. The process of placing a new replica on a mobile node (replica allocation or placement process) should be complemented by another process, which deallocates no longer needed replicas. In this way, the replica distribution can be adapted in the network and preserves its resources.

3.3.2. Selection

Selection of one mobile node (leader) from a set of alternatives to be responsible for delivering a specific functionality inside its partition is a common issue in MANETs. The selection of a leader is known as the "leader election" problem. Leader election is very important for managing a set of concurrently running replicas in a given situation. Leader election can be achieved based on some criteria like mobile node stability, available resources, locations and the network topology features. On the other hand, from a client perspective, a service invocation can be understood as a selection process as well, in which mobile clients are selecting one out of a set of reachable providers. The following questions can describe generally the role of the service invocation in the selection: How can a client find a service provider? What should it do when it looses its provider? Can a client continue from a specific point of execution when switching between two providers? If the service can be cached, when is it allowed for a client to restore it from its cache and publish it for the other network participants?.

3.3.3. Synchronization of Replicas

It is should be kept in mind that it will not be always possible to preserve a set of replicas consistent and synchronized inside a MANET. Regarding the partitioning behavior, two network partitions may remain separated forever each with its own replica which operates under different sequences of requests or transactions. In this case, while replicas are in separate partitions, no recovery mechanism can be applied to keep these replicas consistent [DJ07]. Moreover, the global consistency of replicas is not required by many applications [HM09]. [HM09] considers the consistency maintenance based on local conditions such as location and time and introduces a classification for these consistency requirements. How are the update messages exchanged between the different replicas? How can the states of a service be synchronized? The previously mentioned questions are very important to be answered by any replication approach for services in MANETs.

3.4. Classifications

Many classifications have been introduced for the replication approaches in distributed systems in general like in [WPS+00], data in peer-to-peer environments like in [BDS07], and data in MANETs like [DB09], databases in MANETs like in [PGVA08], and services in MANETs like [HDKR10]. We draw our thoughts about

how to classify the data and service replication approach in MANETs from a service replication perspective.
Based on the previously mentioned issues of data and service replication in MANETs, the data replication approaches can be classified into two main classes. The first class is the network status aware protocols. In this class the replication approach is designed to keep the data or service availability as high as possible by analyzing the current network topology, connection status, partitioning behavior, and movements of the mobile nodes. The approach replicates and allocates replicas upon detecting some event occurrences in the network. These events can be the detection of the partitioning behavior at/by some (or groups of) network participants, long paths or critical connections between the nodes. Moreover, we include any required classifications for the mobile nodes by the replication approach as knowledge that requires awareness about the network status. We call the first class the *network status aware* replication approaches. The second class contains the approaches that do not pay any attentions to the changing the network status. Instead, these approaches use other effects to trigger the replication process in MANETs. We call this class the *network status transparent* replication approaches.

3.4.1. Additional Features

Topology awareness is our main criterion which classifies the replication approaches into two classes. The approaches in these classes have different features. Each of them covers some key issues. These features can be mapped to the energy issues of Subsection 3.2.2, scalability issues of Subsection 3.2.3, managing replicas issues of Subsection 3.2.4, and service awareness issues of Subsection 3.2.5. The introduced issues and features can represent the motives behind designing the replication approach in a MANET. So, the replication approach can be a topology-based scalable push-based approach or a topology unaware service-aware energy-aware replication approach.

3.4.2. Network Status Aware Replication Approaches

Network Partition Aware Replication Approaches

Chen et al. [CN01] proposed an integrated data lookup and replication scheme to improve data accessibility in MANETs. Based on the concept of group data access, a group of nodes collectively hosts a set of data for all. In such concepts, a set of mobile nodes forms a group. The formed groups host collaboratively a set of data items to increase their accessibility. A peer-to-peer data advertising protocol is applied to exchange the information about the available data items and resources between nodes. Each node periodically broadcasts *ad* messages to the group using an adaptive sending policy. These messages are used to establish the required information advertisement about the availability status in the network partitions. The advertised messages are used by other nodes to reduce data redundancy within a group, and to derive a local data lookup table. The ad message contains an identifier, the available data items at the node, and available resources at the node. When a node receives an ad message from another node, it updates its knowledge about the availability through its lookup table for its own use. Deploying replicas of these data items is done between the mobile nodes to increase the data availability. By gaining the information of the availability from the other nodes' ad messages, redundancy of replicas can be avoided. Each data item has a time stamp which is used in deletion of older data items' replicas when newer versions are available. The proposed replication scheme predicts group partitioning based on the nodes' current locations and movement patterns, and replicates data from one partition to another before partitioning occurs. This proposed algorithm detects the partitioning behavior of the network by using location based routing protocols like in [AB08]. It uses the Global Positioning System (GPS)[2] to predict the movements of the members of the group and tries to deploy replicas of the data items in the partitions which are ongoing to be formed. Once the occurrence of a network partition is predicted, the replication process is done. The new replicas are allocated on the mobile nodes with the most resources. These nodes can be selected based on the ad messages.

Wang et al. [WL02] introduced a partition prediction service coverage algorithm for MANETs. Once a partitioning behavior is detected, it tries to push a replica on the farthest node in the departing partition. In case of presence of two or more servers in the same network partition the server with the eldest ID is allowed to continue. Using the relative speed, a client should search for reliable providers in terms of similar relative speeds. Beside, the presence of a suitable routing protocol and GPS, the authors also assume the existence of group mobility patterns in the network and use these patterns to predict the network partitioning. A set of algorithms to perform the velocity analysis is proposed. A reference velocity group mobility model (RVGM) is introduced in [WL02] which is an extension to the reference point group mobility model (RPGM) [HGPC99]. RVGM identifies a certain number of mobility groups with the references that define their mobility behavior. The servers are supposed to know the location information of their clients and apply a clustering prediction scheme to compute proactively the future partitions. The replication process should be delayed as much as possible to reduce the effects of prediction errors. Clients should rank their servers and pose their queries to the most stable server which is in the same mobility group and with similar velocity

[2] `http://www.gps.gov/`

characteristics. Clients are asked to update their favorite servers periodically to allow them to stay connected to the most stable servers as much as possible.

Derhab et al. [DB07] proposed a service replication protocol, PSRP, that can predict network partitioning before its occurrence. The protocol constructs a directed acyclic graph (DAG) that is rooted at the server node. The protocol uses a TORA [S. 99] partition detection mechanism that helps nodes to detect if they still have a path to the server. To predict network partitioning, the protocol classifies links into strong and weak based on their residual lifetimes. A node that loses its last strong outgoing link towards the server, executes a diffusion computation in order to find an alternative route composed only of strong links towards the server. If no such a route is found, the node concludes that the server node is about to be unreachable. In this case, it downloads the service and becomes a passive server. If there are no servers in the new predicted partition, the passive server becomes an active server. If there is more than one active server in a network partition, a merging algorithm will be triggered. An enhanced version of PSRP, SSRP [DB08a], is proposed to work in a highly dynamic topology. When the topology changes at a high rate, multiple diffusion computations can be triggered concurrently. In this case, PSRP chooses to stop the oldest computations in favor of the most recent one, which means that PSRP restarts execution from scratch whenever a new topological change occurs. If topology changes do not stop, PSRP will not terminate. Thus, network partitioning might never be predicted or it is predicted after the partitioning has already occurred. To fix this shortcoming, SSRP changes the criterion used by PSRP to order computations and proposes to stop newer computations in favor of the oldest one.

Dustdar et al. [DJ07] introduced a web service replication protocol for mobile services. It depends also on an analysis for topological changes and selects the best node for hosting a new replica. The nodes are informed about the whole network status through an installed replicator component. The proposed replication mechanism requires: (a) The "monitor" component which checks periodically for each change of network status and spreads the information about that. The most powerful nodes (in terms of available resources) are selected to be the monitors. "Monitors are responsible for partitioning the nodes by their IDs into groups and detecting changes into these groups" [DJ07]. Moreover, they are able to merge these groups into a new group. (b) The "replica placement mechanism" which keeps track of the replica status and if it needs to be corrected. Each client should find a primary copy of the service for its requests based on the available resource. A related leader election based process on the available information about the mobile nodes takes place to determine the suitability of a node to host a new replica.

(c) For service synchronization, it assumes that lazy background updates for the concurrent replicas will be sufficient. An internal database to keep track of the global status of the network is required. The protocol suggests service hibernation based on monitoring and analysis of the network status. (d) Service invocation is done through a standard web service description protocol (WSDL). A related WSDL-finder component keeps track of the changes of the offered services in the network and provides valid WSDL files for the clients.

Summary: The approaches describe above in Subsection 3.4.2 introduce a set of data and service replication protocols in MANETs. These approaches are designed to cover the data and service availability despite the partitioning behavior of the network. The techniques used for the partitioning prediction and detection are variant. Using GPS is applied and suggested in [CN01]. Other replication approaches [WL02, JSHS04, JJKY04] suggest usage of GPS in refining the partitioning prediction processes. In [WL02], analysis of the mobility patterns of the nodes can provide some judgments for the partitioning behavior. In [DB07, DB08a], the partitioning behavior can be predicted based routing protocols. Using a suitable routing protocols to capture some topology features or a whole image of the links of the network is suggested also by [WL02, DJ07]. The previously mentioned techniques are the most notable techniques for predicting the partitioning behavior of MANETs. Although some of the approaches described above can be considered as partition aware approaches, they can meet other requirements like scalability and power awareness issues like the approaches of [HSC01]. Therefore, these approaches are going to be included and considered for other issues.

Scalability and Replication Approaches

Hauspie et al. [HSC01] presented a replication decision algorithm based on link evaluation. It assumes presence of a suitable routing protocol that can be queried for long routes in dense MANETs. The authors assume that not only the network partitioning behavior is the main actor in the process of service replication but also the long wireless routes and routes with several intermediate nodes, which affect the quality of the delivered functionalities. So, partitioning behavior here is widened to include the low quality links between nodes which may appear broken in terms of reliability, bandwidth, delay, besides lack of transmission power. This approach considers the conditions of the wireless connections as the initiator of the required replication decisions. The algorithm detects these long and critical wireless routes (between a specific server and a client node) and tries to deploy a replica at the closest valid node to the client by employing a set of concepts like robustness of paths and service path quality which are based on the number of essential nodes (essential for a path between two nodes to be valid) in the path. Also a length of path threshold (in term of number of hops) is used in order to compute the robustness of a path between two nodes. The robustness considers the number of disjoint paths and their lengths.

Bellavista et al. [BCM05] introduced an optimistic decentralized middleware solution for cooperative replication in the dense MANETs (REDMAN). It can dynamically determine the nodes that are currently participating in dense MANET. A node is in the dense MANET only if the number of its neighbors is greater than a threshold. The data availability is expressed in terms of replication degree. The replication degree is the number of replicas required for a data item or any resource to ensure a certain availability in the network. The solution is designed for MANETs which are dense enough not to be partitioned. The changes in the network topologies are considered with only lookup queries to the data items to be replicated. The replica distribution process (producing replicas) is done based on the replication degree of each resource at its original location. The replicator component (which is elected: the manager role is likely to be assigned to any of the nodes which are located in a topologically central positions of the network) collects the available information about the resources from its original resources, then it delegates the mobile nodes (with the original resources) to start producing replicas. The replicas are supposed to be distributed at certain distances (hops) of each others. Dynamically maintaining the replication degree is considered. The replicator component monitors the replication process to keep the required replication degree in the network. Special information about the resources' locations in the network is forwarded to the other nodes (clients). To avoid flooding the network with this information about the location of the replicas, only those mobile nodes in between the original resources locations and the new replicas are broadcasted. After the initial replica distribution phase, the replication degree maintenance mechanism works to keep the replication degree decided for each shared resource unchanged, without guaranteeing strict consistency, i.e., it is possible to have time intervals where the requested replication degree differs from the actual number of replicas in the dense MANET. RDM aims at maintaining the replication degree unchanged only by reacting to resource delegate movements/failures [BCM05]. This approach requires performing topological analysis for many of its mechanisms. Moreover, it does not provide any mechanism to determine the optimal number of replicas required for each data item that will ensure data availability throughout the network [PGVA08]. On the other hand, it provides an architecture independent solution. Even if it requires performing topological analysis processes like electing the replicator component, these processes are done from the application layer through assumed discover messages. In fact, these assumed messages decouple the proposed solution from special routing components or network architectures.

Huang et al. [HCP03] proposed a replica allocation scheme to improve the data accessibility (DRAM) . Based on modeling for a group mobility where the mobile nodes follow a leader, several mobile nodes move together. In DRAM, a set of neighbors are required to exchange information about their movements. The mobile nodes are clustered into groups based on their movement behavior. After broadcasting "info" messages, the mobile nodes are classified into two groups by the lowest-id clustering algorithm proposed in [HCP03, GT95]. The nodes withe smallest id become members in the "zone-master" and the other nodes become "zone-members". Each of the zone master becomes a leader for its broadcasting zone. The zone masters categorize the zone members with respect to their mobility behavior into mobility vectors. Using a periodic messaging standard between the members and the master ensures that the zone members can realize weather they are in the same broadcasting zone or not. Once any of the member nodes notices the unavailable master case, a new classification process has to be performed. The replication process included by DRAM is based on the allocation unit weight. This weight is a compound feature of the accessing frequency of a specific data item , time stamp, and storage spaces at the different mobile nodes. The largest weight data item replicas are allocated at the highest frequency mobile nodes. This replication approach has a critical draw back. It assumes that there is a similarity between the movements of some of the nodes. However, if there is a large number of nodes that have diverse movement patterns, the groups formed may be small in size and very large in number. In addition, if random motion of nodes occurs, the group joining and leaving operation may become an overhead in the system [PGVA08].

Summary: A replication approach is said to be scalable if it has the ability to adapt to increased demands in terms of network size and node speed without the need for significant additional overhead [DB09]. Scalability in the previously mentioned approaches is considered from many perspectives. The increasing number of the mobile nodes is one important perspective of scalability for data and service replication. The higher number of mobile nodes may produce a dense network. The major problem in dense MANETs are long paths between clients and servers. Allocating replicas far from the servers to produce better replica distribution is considered in [HSC01, BCM05]. The second perspective is how the approach or protocol can structure this high number of nodes. Presence of node clusters, zones and ranges is considered for scalable replication approaches like [BCM05, HCP03]. [DB09] presents and compares a set of scalable cluster based replication protocols for MANETs [DB06, YMH05, ZSL04]. Some other replication approaches like in [AYS$^+$09] consider building the network zones to increase the service availability for an increasing number of the mobile nodes. These approaches support other replication issues like energy consumption awareness in the network. So, we consider them with some other replication issues as it is going to be shown.

Energy Consumption Aware Replication Approaches

Ahmed et al. [AYS+09] presented a distributed replication scheme for services in MANETs (DAR) to improve service availability with reasonable power consumption. Dynamically, DAR divides the network into virtually disjoint zones. The zone diameter is preferred to be at most 2 hops. The nodes with minimum moving speed are selected as the zone heads. All the zone heads are connected through a virtual backbone. The replica allocation process is done dynamically in the zones depending on the service demand level of each of the zones. By using this zone structure, DAR selects a new replica node according to the topology and number of requests from clients in each zone. It can control the locations and number of service replicas which keeps the network-wide energy consumption as low as possible and improves the service availability. If there is more than one active server, the best server based on server's resources (e.g. residual energy, available memory, number of neighbors, etc.) will be allowed to remain active [AYS+09]. Not all of the zones require replicas. Only those zones with a certain level of client request receive a replica. The service availability in DAR is defined in means of the number of hop counts between a client and an active server. The smaller the distance from a client to an active server, the higher service availability is.

Thanedar et al. [TABR04] introduced an expanding ring replication strategy based on pull-based and push-based information dissemination. The strategy involves the presence of an agent on each mobile device, called the Data Agent. The data agent performs tasks related to the exchange of data between mobile devices. In the pull-based approach, each node advertises to its neighbors only the kind of information it needs or in which it is interested. These advertisements, called interest advertisements, essentially contain a description of the required information. The advertisement is not forwarded to nodes beyond the one hop neighborhood in order to reduce the energy expanded by the nodes, as well as the network traffic load. Considerable energy is saved by other nodes in the network since they do not have to expend their battery power processing interest advertisements for nodes that may be several hops away. Once an interest advertisement is created, it is broadcasted to all nodes within a distance of one hop. Neighbor nodes do not further broadcast the interest advertisement. This is to prevent a flood of broadcast requests in the network [TABR04]. If the request has a time out or expired without response, the node composes a request to the server with the required data item. In the push-based approach, the servers compute, based on the access frequency in a certain interval of time, where to push replicas and to which *capable* nodes. The capability is computed in means of the available memory space and the remaining battery power. The mechanism of selecting the candidate nodes to host replicas depends on the distance (in hops) to the servers. The servers are required to periodically probe the network for topology and capability analysis for the replica allocation process.

Shinohara et al. [SHHN06] proposed a replica allocation method for improving data accessibility with balancing the power consumption among mobile nodes. In this method, each mobile host replicates data items considering their access frequencies, the numbers of their replicas, and the remaining amount of their battery power. If some mobile node accesses a remote data item hosted by another node, it considers it to be replicated to its side. If there is enough memory space, a mobile node replicates this data item. Otherwise, the mobile node broadcasts to its neighbors a data information query packet. When one of the mobile node's neighbors receives the query, it generates a response packet containing its frequency of accessing the data items and if it hosts locally a replica of this item. The mobile node computes (considering some defined weights for the remaining power of its batteries and the access frequency of the data item by the other mobile nodes) either to host a new data item or not. The data items with higher access frequencies are replicated on a higher number of mobile nodes. This method distributes the load of accessing an interesting data item over a higher number of mobile nodes. Therefore, some balancing of accessing the data items on the servers is achieved taking into consideration the remaining power in the batteries of the new servers to be selected.

Summary: In the previously mentioned approaches in this subsection, the energy awareness by a replication approach can be achieved from many perspectives. In [AYS+09], the servers are selected in such a way that the lengths of the paths to servers inside the zones are minimized to save the client batteries. As mentioned before, the nearer servers ensure smaller response time. In contrast, the approaches [TABR04, SHHN06] combine the remaining battery power of the candidate nodes to receive replicas during the allocation process. Moreover, [SHHN06] involves distributing the load of the frequently accessed server and directs the requests for the frequently accessed items to larger set of servers.

Link Quality - Access Frequency Replication Approaches

Hara et al. [Har01, HLN03] proposed two dynamic replication techniques: the *Dynamic Access frequency and Neighborhood (DAFN)* and the *Dynamic Connectivity Based Grouping (DCG)*. In the DAFN technique, the mobile node which hosts data items ranks these items descending order to their frequency of access. Then,

it distributes replicas to the mobile nodes that access these data items frequently. DAFN eliminates the redundancy between the replicas and between the neighbors by the originality (if there are two neighbors with redundant replicas, the more recent replica will be eliminated). If the redundant replicas are generated at the same time, the one with lower access frequency should be eliminated. The space of the eliminated replica at one of the neighbor nodes may be used to host a replica for another data item based on its frequency of access. Although the DAFN replicator component requires to scan the network at least once to enable the replica redundancy elimination process, the technique reduces the redundancy only and does not eliminate it [PGVA08].
In the DCG technique, rather than removing the duplications of replicas of small groups of neighbors, larger groups of mobile nodes are involved in this process. It creates nodes that are bi-connected components in the network [DB09, AH74]. Based on the analysis of the bi-connected components, a group of connected mobile nodes may be split into two or more groups, if the bi-connected component is removed. This technique does not assume failures to the bi-connected component nodes and assume that these nodes are connected with many wireless links to the other nodes in the network. Therefore, the bi-connected nodes are considered as stable and replica elimination should not be done on their sites. DCG requires also analysis for the connections of the mobile nodes to determine the bi-connected components. DAFN and DCG build their replication decision based on the current network topology combined with access frequency of the mobile nodes, then they deallocate (eliminate) redundant replicas. In [Har03] periodical data updates are assumed. Based on this assumption, two extensions have been introduced to the previously mentioned techniques. Improving the data accessibility against periodic and aperiodic updates is considered. Each of the data items is assumed to be periodically broadcasted. The replicas may be invalidated if their nodes can not access an originally assumed replica (leader). The extensions count the access frequency of a data item during a specified time interval. This time interval is defined by the time slots between the periodically required updates. The replication is done for the higher access frequencies data items during a certain update intervals. This enables the previously mentioned techniques to dynamically adapt to the varying access frequency during fixed time slots. In [HLN03] three techniques are proposed to decrease the effects of the partitioning behavior of the network. The techniques are *DAFN-S1: DFAN stability of radio link 1*, *DAFN-S2 DFAN stability of radio link 2* and *DCG-S1: DCS stability of radio link 1*. The idea behind these techniques is to perform the replica elimination process over two mobile hosts only if the wireless links (connections) between them are stable. By using prior knowledge about the node location and mobility like GPS information to deduce the movement speed and direction which is used in predicting when two neighbors can be disconnected, the stability of radio links can be evaluated. These techniques con-

sider that it makes no sense to eliminate two duplicate replicas on two neighbors if the wireless link connection between them is unstable. The stability of a wireless link is computed based on the mobility of two mobile hosts. In the DAFN-S1 technique, if there is a data item duplication between two mobile nodes that are connected through a stable wireless link, one of these replicas should be eliminated in the same way as in DAFN. In DAFN-S2, the value $((1 - B_{ij}) \times p)$ is calculated, which represents the probability that this node will access the data item after being disconnected from the other node, such as: p and B_{ij} denote the access frequency to a data item at a node combined to the stability of (i, j) respectively. If this value is more than a threshold value, the node replaces the data item [DB09, PGVA08]. In the third technique, DCG-S1, the value of B_{ij} is compared to a threshold. If it is greater than this threshold, then the two mobile nodes i and j are connected. As in the DCG technique, replica elimination for duplication in each bi-connected group is done. These groups are formed considering the fact that two nodes are connected only if the link between them is stable [PGVA08].

Zheng et al. [JJKY04] provided a dynamic, adaptive replica allocation algorithm which aims at minimizing the communication cost of the replication process. It combines the two concepts of topological change analysis and frequency of requests (read/write requests) to perform its replication processes. It allocates the produced replicas where it can minimize its access cost. It suggests a usage of GPS or a suitable routing protocol for processes of the topological change analysis. It assumes that the replica nodes (providers) can exchange a write request (transactions) immediately once they receive any transaction that changes the replica's state. The adaptive replica allocation algorithm for MANETs (ARAM) is introduced. ARAM uses the collected requests of a replica neighbors (the neighbor set) to perform dynamically the replica allocation scheme. ARAM makes periodic tests on the collected requests in a varying time interval (t). The shorter the t intervals are, the higher the ability to deal with higher and frequent network status changes is. Stable neighbor is defined as the node that has a stable path (all of the intermediate links are available at a specific time) to each node in its neighborhood set. ARAM is extended to be ARAM with stable neighbor (ARAM-SN) which extends the network status analysis to find stable neighbors to host the replicas. The evaluation of the replica allocation scheme in ARAM-SN is accomplished only by the stable neighbor set.

Summary: The previously discussed protocols represent a representative set of the non-partition aware, non-scalable, and non-energy consumption aware replication approaches. Combining the access frequency of some data items with measurements about the quality of the wireless links can provide a way for achieving data dissemination in MANETs. The wireless link quality can be expressed in different meanings. It can be the "residual" link lifetime as needed in [DB07, DB08a], or the stability of wireless connections as mentioned in [HLN03]. Producing replicas to decrease the access cost can be achieved by the analysis of the access frequency combined with the analysis of the links as mentioned in [JJKY04].

3.4.3. Network Status Transparent Replication Approaches

The data replication approaches for MANETs discussed so far share a common feature. All of them require to deal with some measurements and features which are extracted from the network status. These protocols are supposed to be network architecture dependent (Except REDMAN [BCM05], although it requires performing network status estimation operations like sensing the dense network places and estimating the number of hops to the farthest node). Including REDMAN, all of these protocols maintain information about the network status. On the other hand, some replication approaches can increase the availability without being in need for the network status information. In this subsection a set of examples of these approach is presented and discussed.

Flooding and Data Dissemination in MANETs

Most of the data dissemination protocols are based on flooding techniques. [BBHS05] introduces a set of pull based and push based data dissemination approaches. [DB09] summarizes and presents a set of flooding approaches to increase the availability as in [HP04, HHN06]. In [WC02] various flooding protocols are presented and classified. Selective flooding of data dissemination in MANEts is presented as a technique with low overhead. Epidemic strategies are used in the selective flooding approaches. In these strategies, the mobility of the network participants is used in spreading the data in the network [BBHS05]. The effects of the mobility and the query patterns on data dissemination in MANETs are presented in [PS00]. The considerations for power consumption are introduced and investigated in [PS01]. In [BBHS05], the network load has been considered in a pull adaptive based dissemination technique for MANETs. In [KCC05], more concerns about the mobility of the mobile nodes are considered. The effects of more realistic and heterogeneous mobility patterns are summarized with the data dissemination approach of [BBHS05]. On the one hand, the flooding strategies consume aggressively both network and mobile nodes' resources. On the other hand, the selective flooding strategies reduce the number of required messages for the flooding and number of the required nodes for the dissemination. [HHN05] considers the cached data invalidation, proposes an updated data dissemination method that reduces the number of access for the old replicas, and updates these replicas efficiently. However, neighbors are selected randomly or based on any other selection strategy which leads to that some nodes in the large network may not receive the messages or may not be able to find some of the data items at all [DB09].

Access Frequency

Hara et al. [Har01] proposed three methods to achieve data replication. We consider here the first replication method which is a *Static Access Frequency (SAF)*. In this replication technique each mobile node ranks its data items in a descending order of the frequency of access. Then, it allocates replicas in the mobile nodes that access these data items frequently. There is no duplication control mechanism where several replicas for the same data items are allowed to be allocated on neighboring nodes. Although in [Har01, Har03], it is stated explicitly that these techniques can estimate the network connection status, we consider that this can be done to the one hop neighbors without being involved in any network, topology or routing analysis processes. In [Har03], it is assumed that the access frequency for a data item can be collected periodically at a fixed certain time interval which required to maintain periodic updates to the data items.

Our Contribution

The main contribution of this thesis is a service replication approach for MANETs. The approach is called the *Service Distribution Protocol* for MANETs (SDP). We are motivated to indicate the position of this approach to the proposed classification. As we are going to describe in detail during the next chapters, it is an architecture independent approach which requires manipulating information about the service popularity or degree of importance for a certain set of clients at a certain time. It does not require any network topology estimations or analysis. Moreover, it does not require any probing for the network partitions or participants. The service popularity information can be extracted from the application layer. SDP is based on not only the access frequency but also how this frequency is shaped from a set of clients' requesting behaviors (as discussed in Chapter 6). SDP using the mobility of the mobile nodes to let the popular services prevail in the network in an epidemic style. From a classification perspective, SDP is a network status transparent service replication approach.

3.5. Converging to Service Replication in MANETs

The service awareness in the previously mentioned protocols is not clear. In fact, this is due to the fact that a service can be normalized as complex data item to

be replicated as previously mentioned in Subsection 3.2.5. Some of these previously mentioned protocols explicitly propose their consideration of services. [DJ07] proposes a solution for web service replication in MANETs. It considers also the stateful connection for the services. Other approaches like [AYS$^+$09] consider SOA. SDP presents also means for service replication in MANETs. As we are going to show it utilizes the presence of assumed SOA components to provide the replication process. More details about definitions for he service replication as a process and replicable services are introduced in Chapter 5.
[HDKR10] introduces detailed comparisons between some of the previously mentioned protocols as service replication approaches in MANETs. The introduced comparisons are based on the previously mentioned replication issues and approach features. This comparisons can be summarized as shown in Table 3.1. In this comparison, we included SDP. It is abstractly and conceptually compared to the other alternative approaches. In the next chapters, more details about SDP concepts will be introduced.

The important notes out of the comparisons of [HDKR10] and Table 3.1 are: (1) All of the protocols answer the selection questions explicitly except Hauspie's [HSC01]. Conceptually, the replication process in [HSC01] does not consider this as a problem. In a very dense MANET and considering the mobility of the servers, it is very probably that we have two very close (in number of hops) active servers. In this case, selecting one of the active servers will be important to the clients. (2) Although SDP is an optimistic replication approach, it assumes the presences of a lazy background update process for service consistency as assumed in Chapter 5. SDP also allows the set of clients in a certain network partition to dynamically elect their provider nodes based on the service interest, where the uninteresting (with low popularity) services will not receive requests from the client and hences will be shut down as presented in Chapter 4. Neither Hausepie's nor Wang's show their consideration regarding this important issue. (3) All of the protocols require a certain set of requirements or infrastructure to be functioning except SDP. We extend this comparison criterion to include the REDMAN [BCM05] approach as an architecture independent approach. As previously mentioned, it requires performing topological analysis to determine the extended ranges. (4) Regarding the presented classification in Section 3.4, half of the protocols are push-based and half of them are optimistic replication protocols. All of the protocols are either partition-aware or topology based approaches except SDP. (5) PSRP and SSRP show a complete set of solutions for service replication in MANETs. They provide typical well described partitioning behavior prediction mechanisms based on TORA routing component with a pull-based replication scheme. PSRP and SSRP execute the partition prediction only when some topological changes occur and not periodically and this behavior enables them to produce less complexity mechanisms for replication. As proved in [DB09], using analytical studies, both of PSRP and SSRP showed the

		Hauspie et al. [HSC01]	Wang et al. [WL02]	Zheng et al. [JJKY04]	Dustadr et al. [DJ07]	Derhab et al. PSRP and SSRP [DB07, DB08a]	SDP
Replication Decision	When?	On long connections to a server in terms of quality are detected.	On partition predicted	On higher read-write and topological changes	On topology of network status changes	On partition predicted	On high interest detected by a client
	Where?	A closer node to the client.	Most far distance node in a departuring group of nodes	Neighbor nodes where the decrease in the access cost > the increase of the update cost	Best suitable node in terms of its available resources	Nodes which predicted the partitioning behavior	Interesting node
Selection?	Leader (S)election? (Multi-Server)	-	An elder-ID selection is carried out	Replicas with requests from replica-nodes higer than the gained request of the common nodes should be relinquished	Most resourced node is elected	Stable active server	Short election mode Long election mode
	Provider Selection? (Invocation)	Nearest servers are selected	In terms of connectivity and relative velocity	-	Primary copy of a replica	Available active server	The most Interesting service
Replica Consistency		-	-	Simply via replica nodes as write requests	lazy background updates	Forces multiple active replicas to be merged	Assumed lazy updates per partition Background updates
Techniqes		Robustness of paths	Predicting the partitioning behavior based on group mobility patterns	Stable neighbor Service access request	Internal Database Monitor Hibernation	Partition Prediction Residual link lifetime	Hibernation Caching Gross Interest
Requires?		Suitable routing protocol	Caching Analysis for velocities of the mobile nodes A routing protocol or GPS	A routing protocol or GPS Available history information for vicinities	Caching Requires a global view for the whole network status	TORA [S. 99] Neighborhood stability	-
Features		Pull-based – Asynchronous Partial – Optimistic Partition-aware	Push-based – Asynchronous Partial – Optimistic Partition-aware	Push-based – Asynchronous Partial – Non-optimistic Partition-aware	Pull-based – Asynchronous Partial – Non-optimistic Topology-aware	Pull-based - Asynchronous Partial – Non-optimistic Partition-aware	Pull-based - Asynchronous Partial – Optimistic -

Table 3.1.: Service replication protocols and their features [HDKR10]

best complexity for prediction replication and query costs compared to the other alternatives. Although [DJ07] also shows a complete consideration of all criteria, it suggests the presence of a network analysis component but without descriptions. Further in this thesis, the performance of SDP is compared to PRSP and SSRP as a set of representative protocols for service replication in MANETs. The main point to be highlighted here is that SDP is a network transparent replication approach unlike the other two protocols which are network status aware protocols.

3.6. Performance: Parameters and Metrics

The previously mentioned issues form the main parameters that affect the service replication process in the MANETs. In this section, we briefly introduce a set of the parameters which have been mentioned as the independent parameters in most in our literature review. Then, we present and classify a set of metrics which can capture the performance of the replication process. These performance metrics represent the dependent performance measurements.

3.6.1. Performance Independent Parameters

These parameters can describe the environmental and behavioral parameters that affect the replication process. The environmental parameters can be summarized as follows:

- The *network size* or the number of the network participants at some time. If the number of mobile nodes in a MANET is too small and the network is placed over a wide area, the network density will be very small. In this case, it will be very hard to be increase even with applying some replication scheme the data or service availability. If network size can make the node dense enough, the network can avoid being partitioned and probably one deployed server can satisfy most of the participant requests.

- The *mobility of the mobile nodes* affects deeply the achieved replication process in MANETs. Not only the assigned velocities for the mobile nodes but also the mobility patterns are important. In Chapter 9, we give more details about a practical study of the role of the mobility in the SDP performance.

- The *frequency of partitioning* and the wireless links' failure model have also a direct effect on the replication process.

- The *number of the intially deployed servers* in the network. A higher number of deployed servers can enhance the replication process and enables the network to reach a better service distribution quickly.

- The *available resources* on the mobile nodes like the processing power and the memory space present an important parameter to be taken into consideration. If the resources at some mobile nodes are not sufficient for a service to be replicated, these nodes are excluded and can not participate in the replication process. Moreover, by considering the resources of the servers and how these resources are utilized by the current requests of the client, not all of these servers can serve further requests unless at least some (or all) of the current requests are served. These parameters have been introduced and discussed for SDP in Chapter 11.

- The *required number of replicas* to ensure some level of availability in some network scenario is defined as the *replication degree*. The replication degree is treated sometimes as independent and sometimes as dependent performance parameter. Here, we present it as an independent performance parameter. The replica distribution process for at least one of the replication approaches, REDMAN [BCM05], is done based on the replication degree of each resource at its original location. Moreover, REDMAN assumes the presence of a mechanism that enables evaluation for the replication degree as an input for the replication process.

The other set of the behavioral parameters are related to how the generated replicas are accessed. Not only the access frequency is important but also the access behavior of a set of clients regarding some service or replicas. Also, the applied consistency scheme has an important role here as a behavioral parameter. Some of the invalidated replicas regarding the consistency and synchronization constraints are affecting the availability of the generated service distributions.

3.6.2. Performance Dependent Parameters - Metrics

The efficiency of applying a data or service replication approaches needs to be judged. Quantifying the performance versus the previously mentioned performance independent parameters may reflect the advantages and disadvantages of any proposed replication process. These metrics can be classified into two main classes. First, the general class which indicates a set of general metrics that can be used in analyzing any approach. Second, the specific class which contains a set of metrics for specific features of the proposed replication protocols. The general class can be summarized as follows:

- The *availability* is the main performance feature to be captured by the replication approaches. The motivation behind data and service replication approaches is always to increase the availability. The availability is introduced with many meanings by the different replication protocols. For example, in

[DB07, DB08a] it reflects the average accessibility of the servers by the mobile nodes in a network. It is the ratio of number of the mobile nodes which can access the service or one of its replicas to the whole number of network participants. The availability is not only determined by the accessibility but also by the durability of the accessibility or the survivability. The mobile node servers which lay on limited power resources can not stay functioning forever. We highlight here the term of the general availability which is used frequently in the next chapters of this thesis. The general availability highlights the accessibility of the service. Other protocols like DAR [AYS$^+$09] combine the meaning of the availability to some other features like the number of hops to reach a server. The length of the paths to a server affects the delivered quality of accessing some services. Long paths with numerous hops may suffer from higher failure rates and long delays. In this case, the cost of accessing the reachable servers is assumed to negatively affect the availability. SDP introduces another definition for the service availability, where it ranks the services using their popularity. The more popular services or replicas are, the longer is the time that their servers can stay as active servers. SDP availability reflects the time that a service or one of its replicas remained active to the whole network operation time. Another term used for service availability is service coverage. In [WL02] the service coverage is introduced as the defined general availability.

- The *replication cost* can be defined as the number of the actively deployed servers in the network at a certain time. The *replication degree* as previously introduced can be either a dependent or an independent performance parameter. Here, we consider the replication degree as a dependent performance parameter. The replication degree is another face of the replication cost. In this thesis, the replication cost is introduced as the *prevalence* ratio. The term "prevalence" is used to describe the service distribution process in Chapter 4.

- The optimality of the generated replica distributions is very interesting to be measured. The prevalence ratio or the replication cost can only show the number of servers with replicas inside a network situation. Capturing a measurement about the way that the servers are distributed inside the network is also important. This thesis introduces in Chapter 7 a set of replica allocation or placement correctness computation methods (ratios). Relative to an optimum assumed replica distribution, any other distribution can be compared. The *allocation correctness ratios* can measure how near some distributions are to the assumed optimum distribution. In Chapter 11, more considerations for these correctness ratios in conjunction with a resources perspectives are introduced.

- The *data consistency* reflects the probability of accessing the last updated data item by the replica requests [DB09].

The second class of parameters includes the following metrics:

- The partition-aware replication approaches which use some partitioning prediction mechanism like PSRP and SSRP [DB07, DB08a] introduce a metric like the *prediction error* to reflect the partitioning prediction correctness of the approach.

- The *request cost* indicates the number of hops a request needs to reach the server. The query cost is related to the replication approaches that apply some network topology analysis. Considering the network density, link status and traffic, and capacity of the intermediate nodes' buffers, the query cost can explain with the availability the expected quality of the overall replica request-response process in the network. From a data consistency perspective, a request may be a query (read), or write (update). [DB09] distinguishes the request cost into query cost and update cost parameters.

- The response time is introduced to capture an important feature for the service execution in general. In MANETs, the service response time plays an important role in measuring the quality of the delivered functionalities based on the advertised functional and non-functional attributes of the services. The response time can also be considered as a dimension for the availability. Availability may be expressed partially by the response time if there are any time constraints on the received response of a specified service.

- The number of emitted packets in [HSC01] and the network traffic [DJ07] have been used to indicate the effects on the other network functionalities by the replication process.

Most of the performance parameters and metrics which have been used in this thesis are collected and presented with their definitions as an attached glossary by the end of this thesis.

3.7. Other Related Background

3.7.1. Replication versus Caching

Not only data replication techniques may increase availability. Other techniques like data caching increase the availability as well. Data items may be cached in sites away from their original site in the network using the same replication mechanisms. Although replication and caching may look similar, [DB09] introduces a set of features to highlight the differences between the two techniques where the caching is defined as a "byproduct" of a client data querying process. These differences can be summarized as follows: The cached data items are kept by their hosting node independent of any consistency or synchronization constraints until they are explicitly deleted. On the other hand, keeping the replica consistency and synchronization plays an important role in the replication processes. From an availability perspective, while deployed replicas out of a replication approach increase the availability and accessibility of some data or services to a set of clients which may be located away of the original servers or leaving to join another network partitions, the cached replicas increase the local availability for the caching party or node. As we are going to present, caching replicas is one of the core mechanisms of SDP. Some of the network participants may cache some replicas as mentioned in Chapter 4 and 5. In Chapter 10, a method to generalize the local availability of the cached replicas is presented.

3.7.2. Availability as a QoS Component for Replication

In [On04], one of the main objectives is to consider the service availability as a QoS parameter which can be tuned for the replication processes in wired networks. The concept of Quality of Availability (QoA) is introduced to enables the treatment of availability as a controllable and observable QoS parameter. It is shown that the users can specify their service requirements in terms of QoA and how these different QoA requirements and guarantees can be managed and mapped into the low-level replication specification.

3.7.3. SOA and Service Discovery in MANETs

As mentioned in Chapter 2, realizing SOA in MANETs represents an interesting related background for our work. The challenging features of MANETs make the implementation of the wired network SOA structures like the service repositories infeasible. For example, centralized service repositories in face of the ever-changing topology and the network partitioning behavior can not be applied. Service discovery is one of the main principles of SOA. It includes a set of processes and functionalities. In [VP08], service discovery is defined as a process that allows the different network entities to:

- Advertise their services.
- Select the most "matching" service to pose the service requests to.
- Interface to the selected service for invocation and execution.

Since we assume presence of SOA components like service repositories and rely on an assumed service discovery approach (see Chapter 4 and 5), presenting a set of examples for the approaches that can realize deploying SOA in MANETs is considered here. Many approaches have been contributed to cover deploying the SOA components to support the whole service discovery processes. A classification for these approaches in MANETs is introduced in [VP08]. [GPVD99, FKM06, Jus05, Han05, KKRO03, Kle04] represent a set of the work that consider the MANET feature in SOA implementations. In Chapter 4, more details about one of these approaches are given (DIANE model [dia]). Moreover, the interactions between this model and the proposed service replication protocols are presented and discussed.

What has been learned from the state of the art? In the first place, the issues that influence the service availability in MANETs. The service replication dimensions and how the different approaches can achieve it. The special awareness needed in the service replication in MANETs. Based on the introduced classification in Section 3.4 and the comparison of Section 3.5, most of the service replication approaches are network status aware approaches. They need special network analysis processes. These analyses are expensive and makes these approach in need of presence of other special network components. Moreover, as a disadvantage, these approaches do not consider that MANETs consist of different participants with different platforms. They propose solutions for network nodes with similar specific architectures which is not guaranteed in reality.
SOAs have a set of interesting features which fits for MANETs and can come over the disadvantages of the most of the service replication approaches. They support managing and sharing the services in heterogeneous environments. Supporting these different platforms for sharing the functionality seems very promising in MANETs. Exploiting the available informational about the services and the SOA components may provide better information to be used in providing the replication decisions. In this thesis, we want to identify new sources of information for the replication process that can avoid the coupling to specified network architectures and components based on SOA components.

3.8. Summary

In this chapter, we introduced our literature review and the state of the art for the related background of our work in this thesis. The replication for data and services as a solutions for increasing the service availability in MANETs is presented. The issues that affect the data replication in MANETs has been provisioned and discussed. We highlighted the distinguishing features of service replication and the required awareness for the service replication in MANETs. Afterwards, a service model and the important feature of the service replicability have been introduced. The replicability of the service represents a main question for the service replication process. We discussed the design requirements in general for data and service replication approaches from a MANET perspective. A proposed classification for the different replication approaches has been introduced and we highlighted the position of our contributions in this classification. Then, we tried to address some answers and guidance for the important question of how to capture and quantify the performance of a replication approach. Finally, we provided a brief introduction for literature of the other related background. Based on the previously discussed concepts and principles, we present in the next chapter our basic ideas, principles, assumptions and proofs about our contribution of the service distribution protocol (SDP).

Part II.

SDP: Service Distribution Protocol for MANETs

CHAPTER 4

SDP Basis

"Nothing comes out of nothing"

(Descartes)

SOAs added new concepts for MANETs. The work of this thesis exploits the presence of SOAs in MANET and their offered capabilities to provide and guarantee a higher service availability. As presented and discussed in Chapter 3, the process of service discovery is responsible for organizing the advertisement, selection, and invocation of services. One important feature is achieved by the discovery process is collecting the possible information about the services in the nodes' repositories. We aim to utilize this information to provide better service replication decisions to enhance the service availability based on the needs of specific clients. Thus, the work of this thesis complements the presence of well defined and functioning service discovery standards of any applied SOAs for MANETs. The objective of this chapter is to propose and evaluate the required service replication mechanisms based on the reduced information about the services in the distributed service repositories. Furthermore, in this chapter, we estimate the integrity of our proposed replications mechanisms to SOA examples and and under which assumptions.

The main questions that needed to be answered when designing a service replication scheme are : When to generate a replica? Where to allocate the generated replicas? When to invalidate and deallocate a replica? How will clients choose the suitable replicas to communicate with? How to manage replica consistency? These questions have been introduced and discussed in Chapter 3. In this chapter, the main concepts and ideas behind the service distribution protocol (SDP) which are initialy set to answer the first three of the previously mentioned questions are introduced. We make some simplifying assumptions regarding the other questions which are going to be considered as motive for other parts of this work. SDP uses (principally) two mechanisms: The *replication* mechanism for producing and allocating a replica based on the gained service interest by a client (this client is interested to receive and operate a replica). On the other hand, the *hibernation* mechanism is responsible to deallocate replicas from provider nodes. Deallocating the replicas of a provider side is triggered by how this provider losses the interest of keeping a replica allocated at his side. Moreover, an example for the SOA service middlewareand how SDP can use its components are presented and discussed. Afterwards, accordingly, the general assumptions, requirements and conclusions are drawn. The work presented in this chapter has been partially published in [HKR08c].

The structure of this chapter is as follows: in Section 4.1, the basic ideas and principles which will be used by SDP are presented. The two main proposed SDP mechanisms, replication, hibernation and caching, are introduced in Section 4.2. The presented SDP mechanisms are tested using a simple network model which is described in Section 4.3. In Section 4.4, a detailed simulation for SDP is used for evaluations and the results are presented, analyzed, and discussed. As an example of how can SDP integrate to a SOA model, the DIANE model is introduced in Section 4.5. In Section 4.6, The interactions between SDP and The DIANE components are presented. A set of general assumptions which are needed to operate SDP are introduced and discussed in Section 4.7. Finally, the work of this chapter is summarized, the contributions are highlighted, and the next research motivations are presented in Section 4.8.

4.1. Basic Ideas

Consider the following very simple scenario: We have an ad hoc network where just one service is offered. To ensure the availability of this service on all nodes, we could simply make a copy of the service available on each node, i.e., replicate the service to every node. Now, regardless of how the network changes over time, the service will be available to each node. Unfortunately, the price to pay is rather high: Replication itself requires some effort and running the service will deplete each node's resources. Thus, we should somehow restrict the replication. Then, a number of reasonable factors come to mind: First, there might be nodes that will never use the service. Why should they keep a replica? Second, there may be nodes where lots of replicas are in the neighborhood. Again, why should they keep their own replica? Thus, obtaining replicas should depend on the interest a node has in the service. This is what our replication mechanism does. Since a node's interest in a service may vary over time, it should be possible to get rid of replicas that are no longer wanted or needed. This is the target of the hibernation.

Both client and provider service interest to replicate or hibernate some service could be a combination of many features like service publishing time, calling frequency, number of connected sessions (load), and the pre-requirements of this service. The client interest in some service defines how much a client is interested to host a specified service. The higher the interest it has, the higher the possibility to receive a replica. The provider interest in service measures how much the provider does like to keep a service active. The lower the provider interest the lower the possibility to maintain a replica. So the interest is associated with the replication and hibernation mechanisms.

Service vitality for the network participants is determined by their interests. This interest is not supposed to be constant during the whole network operation time. So, the dynamic changing interest of the client should be considered by the replica allocating (placing) mechanism. Services with high interests (popular) will prevail. Most services will not be vital over the entire operation time, these services will survive and be placed wherever the interest (popularity) is high enough, otherwise they will be hibernated.

Using only the service interest measurements can provide the required replication decisions. avoid many t predicting the affecting topology changes of the network. This feature can avoid decouple the replication decisions and the other network components (like the routing protocols as discussed in Chapter 3).

4.2. The Main Mechanisms of SDP

The main actor of SDP is the service interest or popularity. Many criteria can be used to identify the service interest like the requesting frequency, opened sessions (load), service publishing time, service prerequisites, last time of requesting, and context. At this stage of the work, the service interest is presented in terms of the requesting frequency which indicates the number of the requests or queries regarding a service during a predefined time interval by service providers or clients.

As shown in Figure 4.1, a service is already deployed on an original service provider which pushed a service offer about the hosted service in the service offers' repository. As discussed later in this chapter, since SDP is a SOA based replication protocol, it assumes presence of service repositories. These repositories are requierd to be distributed in the network as shown. Client A queried the service offers' repository and got a match with the service offer and started using the service. After a while and based on the client A's request behavior, replica R_A is forwarded to client A which turns into an active service provider A. Provider A commits a service offer about its hosted replica R_A into the service offers' repository. After a while, client B is involved in the same process of service requesting then getting a service replica and publishes a new service offer. So on, that the service prevails the network and a service distribution is produced. Depending on the mobility of the network participants these three replicas will either leave the current partition and join a new partition as mentioned before and the service prevail the whole network partitions in the same manner or more than two replicas will remain in the same network partition. In order to decrease the number of remaining replicas, a service ranking process ranks the different replica alternatives. Based on the ranking process, the clients direct their requests to the highest ranked replicas (the most interesting one). Since, the less interesting replicas gain fewer requests, SDP enforces their providers to hibernate these replicas. So, the hibernation process is very important to save network resources and hence to avoid higher replication cost.

4.2.1. Replication mechanism

Upon a replica request of a client which achieved the required replication threshold during a certain time interval, the service replication mechanism is triggered. A service provider is responsible to copy (forward) a replica to the interested clients. Therefore main functionalities of the replication mechanism take place on both service provider and client sides as following:

- On the provider side:

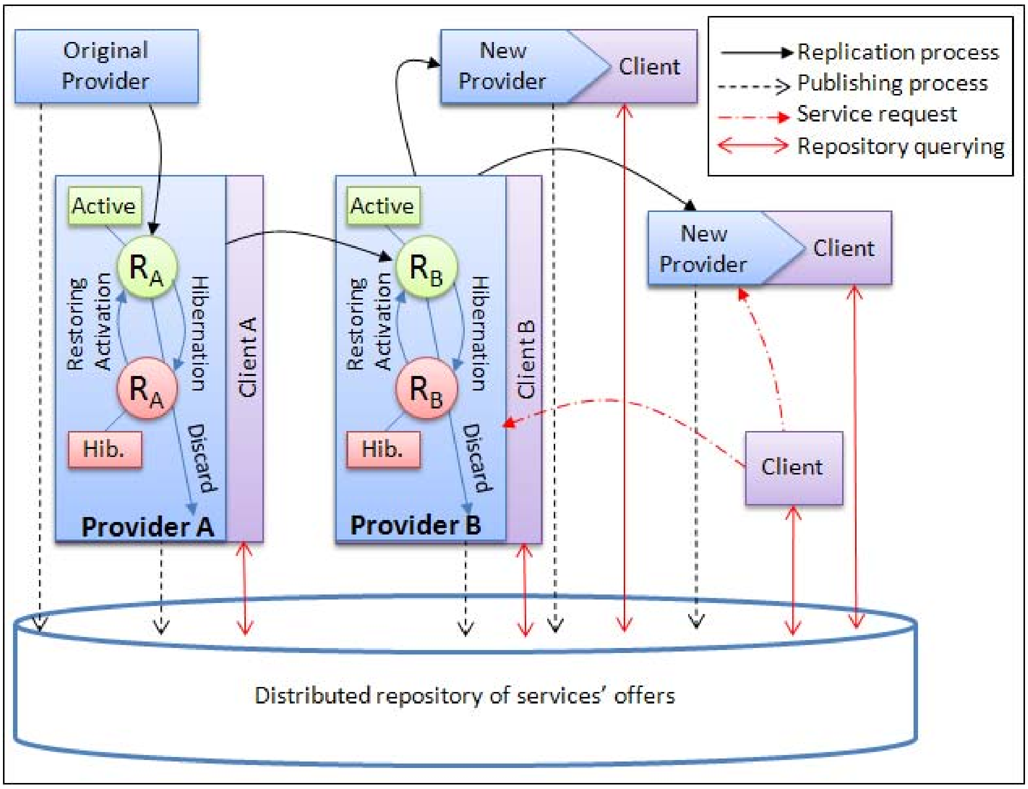


Figure 4.1.: SDP Main core mechanisms

– Pass a replica: Upon a client request, a specified provider passes a replica of the currently running service and keeps information on where the replica has been passed to. The following constraints should be satisfied:

  * The required resources for the service to be run should be available (and/or) transfered to the client site(new provider).

  * The required permissions for accessing any remote resources should be given to the new provider.

  * Any required synchronization for the service status between the new received replica and the other replicas in the same partition should be performed and any consistency issues should be resolved among the concurrently running replicas in the same network partition.

– Publish: Publishes the new status of the hosted services.

- On the client side

- Find the most interesting service: The alternative replicas of the same service are ranked and indexed. Clients find the highest ranked service and try to communicate with it. The popularity-based ranking process for the replicas with same functionality is an important assumption required to enable SDP (as introduced and discussed in Chapter 6). The other important assumed and required component here is the service discovery as mentioned in Chapter 4.
- Count and ask: Counts the requests regarding a specified service and asks for its own replica, if it achieves a specified number of requests (replication threshold).
  The replication threshold represents an indication for a service client to participate in the service replication process. In our work, the replication threshold is based on the general requesting model, shown in Chapter 6, and utilizes only the number of requests in a certain given interval of time. The replication threshold differs from client to client and can be set individually based on a client service-use profile or scheme.
- Switch into a provider: Switches the client to be a provider after receiving, activating a replica and its required local/remote resources and permissions, and publishes the current replica status.

4.2.2. Hibernation mechanism

The hibernation process complements the replication process. It prevents the network from having many unnecessary concurrently running replicas. The replica is hibernated once its provider does not receive sufficient interest in it by the participating clients over a certain time interval. The main functionalities of the hibernation can be described as follows:

- Count and decide: During a specified time interval, this functionality counts the number of requests to its provider by the clients. If this number is smaller than the required number of request in the hibernation threshold, it hibernates this service.

- Cache it: Enables the service provider to cache its hibernated service based on its available resources. Caching the service may include caching just the service executables, like the *.jar* files or also the other required recourses and connected sessions.

- Publish: Publishes the new status of the deactivated replica.

- Discard: Deletes the active or hibernated service. This functionality can be used if there is a new version of the services and the cached one is older.

Both replication and hibernation thresholds are based on the number of posed or gained requests by both of service client and provider only.

4.2.3. Service Caching and Restoring

Service caching is introduced as a subsidiary mechanism. It uses a combination of the functionalities of the replication and hibernation mechanisms. Due to the meaning of the hibernation, it does not involve deleting the hibernated service. This adds some kind of replica caching abilities to the mobile hosts. Caching replicas can increase the local availability of their functionalities at the caching nodes. Based on the core functionalities of the replication mechanism, the mobile host can restore a previously hibernated service. In case of restoring a cached replica, the local and remote resources required by the service should be mounted. Then, a new service offer for the restored service should be published. Restoring a cached replica by a node requires achieving the replication threshold and comparing its Requirements' Index (see Chapter 6) to the other active reachable replicas if any. Chapter 10 presents a set of considerations to generalize the local availability of the cached replicas at the other nodes. The service caching functionalities are:

- Restore a service: Restores a hibernated replica from the cache when the current node (as a client) achieves the replication threshold instead of activating the replication mechanism. The required resources and their permission should be mounted for the restored service.
- Publish: Publishes only the new (un)hibernated status of the restored replica.

4.2.4. Synchronization

Services can be either stateless or stateful. In the stateful services, states affect the service responses as mentioned in Chapter 3. Some services (like the web services) have an internal state, which is desired to be synchronized within all replicas [Jus05]. As mentioned in Chapter 3, it is infeasible to keep a service and its replicas synchronized in a MANET as the network may partition frequently. There is no guarantee that these formed partitions will always be merged. So, SDP assumes that during the sequence of the replication, hibernation, caching, and storing of replicas the active service providers reachable to each other can exchange a set of *lazy background updates* as proposed in [DJ07] to keep their services synchronized.

4.2.5. Role of Mobility in Service Prevalence

Since mobile nodes in any MANETs are moving freely, they may leave their current network partition and join another partition. If the moving mobile node has services, the replication/hibernation processes may be triggered in their newly joined partitions. This traversing behavior does not only depend on the service popularity but also on the mobile nodes' mobility patterns. Thus, mobility is one of the main factors that influences SDP performance. More considerations for the role of mobility are given in Chapter 9.

In this Chapter, basic concepts and ideas for the proposed mechanisms of SDP are investigated and proven. A simple network model, where SDP is applied in the scale of only one network partition ("*artificial partition*"), is described.

4.3. Simple Network Model

At this stage of research, the objective is to measure the effect of the interest based replication/hibernation mechanism on one partition of the ad hoc network. This partition is not a real partition. Maintaining a real MANET partition during in fact is not possible. Therefore, we are gaining to assume a set of features which can form this "*artificial partition*". The assumed network model at certain time is an undirected, unweighted graph $G(N, E)$ where N represents the set of nodes and E is the set of edges. $G_x(N_x, E_x)$ represents one of the network partitions, where: $G(N, E) = G_1(N_1, E_1) \ldots \bigcup G_x(N_x, E_x) \ldots \bigcup G_k(N_k, E_k)$, $N = N_1 \ldots \bigcup N_x \ldots \bigcup N_k$, and $E = E_1 \ldots \bigcup E_x \ldots \bigcup E_k$. The network is placed in a square shaped area. The network topology is varying according to the node movements. Each node covers a fixed radius of radio transmission range R, where R is constant over the network operation time and the participating mobile nodes should be uniquely identified.

4.3.1. Mobility Model

The applied mobility model here is the "Random Waypoint" model [LNR04] in which each node moves from its current location to a specified target location with a constant speed. The speed is uniformly randomly selected in [1,15] m/s; to achieve higher mobility we avoided a speed of 0 m/s. After reaching its destination, each node waits for a pause time. The pause time is uniformly selected in [0,150] seconds. A slight modification was applied by adding a "Mobility index", which is a percentage in [0,100]. A higher mobility index enforces higher speeds and lower pause intervals to be selected.

4.3.2. Workload Model

We assume (and we think it is a very realistic assumption) that nodes with a heavy workload will seem to be unavailable in our network partition. Workload is categorized into the values {25,50,75,100}%. Each node remains loaded with the same workload ratio for a specified time interval of [1,5] minutes, after finishing that interval, nodes are supposed to pick a new workload ratio. Both workload ratio and intervals are uniform random values.

4.3.3. Node Availability

The node availability is an artificial measurement that indicates whether the node is available in the assumed artificial partition or not. A node's availability $A(n_i)$, where i is the node identifier, is based on both the node speed and workload ratio. The following equation shows how $A(n_i)$ is computed:

$$A(n_i) = \begin{cases} true & \frac{(100-workload)(R)}{speed\sqrt{Area}} \\ & \geq \frac{Constant}{(\frac{MaxSpeed}{2})(100-MobilityIndex)} \\ false & otherwise \end{cases} \tag{4.1}$$

Where: *MaxSpeed* is the maximum allowed speed to be selected by a node and if $A(n_i)$, then node n is in $G_x(N_x, E_x)$.

4.3.4. Service Model

We assume a rather simplistic service model: The network has just one service at its starting time. Initially, the service is placed on the first available node. Two main assumption are needed here; (a) All nodes would like to participate in the replication (service distribution enabled nodes). (b) The original service itself is replicable. We are aware that this is a rather strong assumption that will not hold in each and every network. However, there are many realistic application scenarios, e.g., all scenarios involving nodes belonging to one organization, many scenarios involving catastrophe management and the like, all scenarios involving the usage of open source software etc., where this assumption holds. We are also aware, that the assumption will not hold for each and every service. There certainly are services that require specific hardware or software environments etc. which restrict their replicability. Again, however, there are many services that can be replicated to arbitrary devices without a problem.
Also, the feature of a service "Requirement index" is introduced, it represents how high the requirements of an offered service on its host to be operated are (more details will be addressed in Chapter 6). Each replica may have a different value of this index, this value is normally distributed about 20% of a general requirement index value. Client nodes are supposed to find those among the available replicas with the lowest requirement index communicate with.

4.3.5. Requesting Model

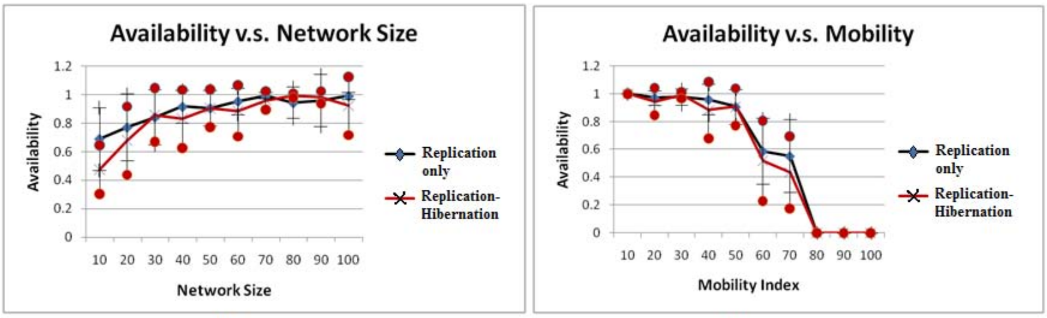
After placing the service on the initial provider node, the available nodes in the network partition become clients of this service. Each client node maintains a requesting rate (number of requests per minute) randomly uniform generated between [0..4] requests/min., a requesting interval which is one of {5,10,15} minutes and a pause interval of {0,5,10,15} minutes. Both pause and requesting intervals are randomly uniform selected.

4.4. Primary Evaluation and Discussions

An extensive simulation for the proposed distribution protocol has been done. A varying number of network nodes (from 10 to 100 nodes) with transmission radio range of 25 meters are put uniformly together in a square area of $300m \times 300m$. The results were obtained from averages of 20 runs, in each run the network operation lifetime was 2 hours. Three main criteria measuring the proposed protocol performance were defined:service availability , prevalence ratio and residence time. These measurements were examined against network size and mobility.
Some common settings were fixed for all runs. The replication threshold, at which the replication mechanism is triggered, is settled to be four calls per minute. That threshold was chosen to be equal to the maximum allowed calling rate. Also, the hibernation threshold is set to be only one call in five minutes. Two groups of results were obtained here: the first group is obtained from just applying the replication mechanism (shown in Section 4.3), the second one is obtained from applying both the replication and hibernation mechanisms. The two groups were investigated against: (a) a varying network size (with a fixed mobility index equals 50%) and (b) a varying mobility index (with a fixed network size = 50 nodes).

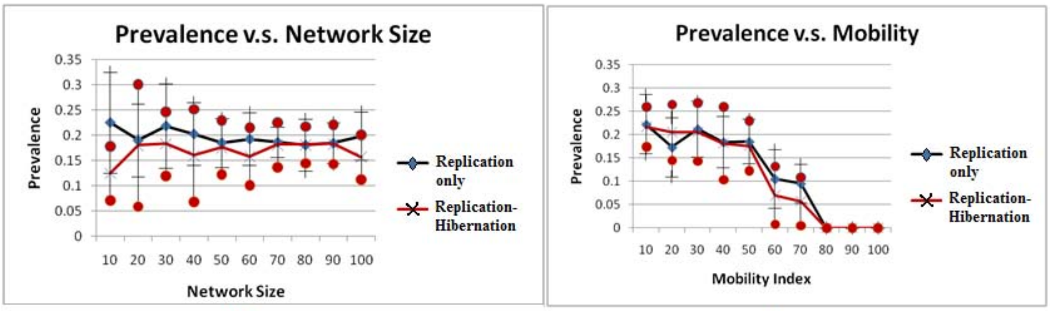
Service Availability:

The service availability is the ratio of the time that the service was available to the total network lifetime. This ratio is very important to measure how much the service (with its replicas) enhanced the availability. As shown in Figure 4.2-A, the service availability increases as the network size increases; this is intuitive because if there is a larger number of nodes this means more requests and more interested nodes to host replicas. Thus, a higher availability can be achieved. Starting from 40 nodes, by applying just the replication mechanism, the availability becomes bounded between 90% and 100%. By applying both replication and hibernation mechanisms, the achieved service availability is always less than with only the replication mechanism. This is due to the purpose of the hibernation, but the advantage is that the proposed protocol decreases the utilization on the network



(A) (B)

Figure 4.2.: Service availability



(A) (B)

Figure 4.3.: Prevalence ratio

links and the service availability does not collapse. The average difference between both resultant service availabilities was always less than 7.5%.

The effect of the mobility on the service availability is shown in Figure 4.2-B. The proposed protocol achieves high service availability for lower speeds (from 10% to 50% mobility index), the availability is bounded by [83..100]%. For moderate speeds (from 50% to 70% mobility index) the availability is about [40..83]%, otherwise it becomes very low quickly.

Prevalence Ratio:

The ratio of the number of nodes that have an active service or a replica during the network life time to the total network size is identified by the prevalence ratio. In order not to waste network resources, like link utilization, a low prevalence ratio together with a high availability is desirable. As shown in Figure 4.3-A, the prevalence ratio converges to less than 20% while the network size increases. On the other hand, as in Figure 4.3-B, the prevalence ratio is affected dramatically by very

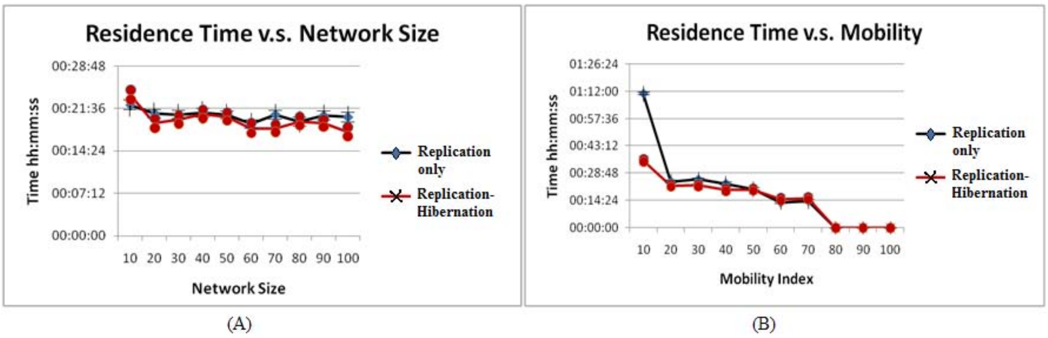


Figure 4.4.: Residence time

high speeds (starting from 70% mobility index). The prevalence ratio decreases slowly between slow and moderate speeds (from 10% to 70% mobility index). The average difference between applying the replication mechanism only and applying the two mechanisms is less than 2%.

Residence Time:

Residence time measures the average time that a service or a replica stayed running on a hosting node. As shown in Figure 4.4-A, residence time is not affected by the network size. By applying both replication and hibernation mechanisms one can make the nodes host the replicas for less time. The average difference between both groups (starting from 30 nodes) of results is 3 minutes. The effect of mobility is shown in Figure 4.4-B, the residence time decreases when the mobility increases and this is an expected result.

Figure 4.5 shows a snapshot of a random run of the simulation and shows the time interval (in horizontal bars) that the different replicas remained active in the network partition. This figure shows the behavior of applying mechanisms of replication, hibernation and storing for a complete run of the simulation. As shown, the service is available for most of the network operation time despite the presence of the unavailable hosting nodes.

The notable conclusions here are:
(A) For the first group of results: applying the replication mechanism of the proposed protocol can preserve high service availability and reasonable prevalence ratio;
(B) For the second group of results: applying both replication and hibernation mechanisms together, which is more restricting and realistic, can preserve a very close performance that is very similar to that of the first group.

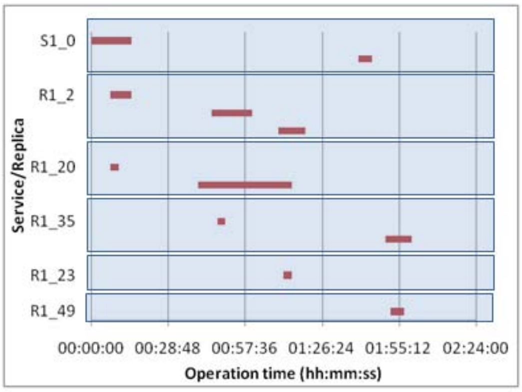


Figure 4.5.: A run shows the concurrently running replicas in the network partition

Further motivation questions:

The main achievement of the previously described primary evaluation is to show the feasibility of using a set of service interest based measurements to control the service replication in the provided network model. At this stage of our work, we have the following questions to be investigated:

1. How can the proposed SDP and its mechanisms interact and integrate to SOAs?
2. Does it remain feasible to apply SDP where a network is ever partitioning?

In this chapter we will look for answers for the first question. The second question presents one of the motivations for the next chapter.

Since SDP is a service replication protocol which assumes that the services are deployed in SOA based environments. The required SOA components to enable SDP to run are needed to be investigated. The following sections discuss how SOA can be achieved in MANETs and how SDP can interact with its components. In the following part of this chapter, the SOA components to realize SDP are presented and showed how to be exploited.

4.5. The DIANE SOA Model

Characteristics of MANETs, i.e., high dynamics, low reliability, limited resources, make it impossible to directly use traditional centralized SOAs in these environments. In particular, maintenance of centralized repositories and centralized management are infeasible [Han06, CC05, HKR06, KT03], since it cannot be guaranteed that a node exists that is: (a) always available and (b) powerful enough to support a potentially large repository and numerous requests for it. Many approaches are introduced to realize SOA in MANETs like in [dia, JDT05, KT03]. DIANE [dia] is a service discovery model that facilitates operating services over MANETs. It is presented here as an example for SOA middleware models that SDP can be merged with. DIANE is a middleware [KKR06] that can manage service publishing, discovery, and composition, as well as invocation and execution. As shown in Figure 4.6, all mobile nodes run a copy of DIANE middleware. Each node may offer an arbitrary number of services. These services are described using the DIANE Service Description Language (DSD) [KKRM05]. Service offers are advertised via an overlay structure[KKRO03] . The overlay used structure (Lanes) has been specifically developed for service discovery in ad hoc networks and can deal well with the dynamics of such a system [KKRO03]. Basically, each node stores information not only about the services it offers itself, but also about the services offered by a number of other nodes. Additionally, a node knows how to get in touch with nodes having different information. Service discovery mechanisms are triggered by the local middleware among nodes in order to find a suitable service. The result of this step will be a list of available services that might match the request. Detailed matching takes place at the requesting node and results in the identification, configuration and binding of the selected service(s). Figure 4.7 shows the basic components of the DIANE middleware. As mentioned before, each of the mobile nodes is maintaining this middleware layer which enables to achieve the required SOA functionalities. The DIANE middleware has three main derivatives to be operated according to the hosting mobile node's role or side(e.g. client, provider or repository). The presented client-side middleware contains a set of agents which enable a client to connect to a repository, search, match and manage services' execution process. Proposing of service offers and operating services is also enabled in this client-side middleware. The repository middleware has the repository and prematcher agents which are responsible for handling registries and filtering and ranking of the offers. The provider-side's middleware most important agent is the OfferAgent which manages the invocations of a service to the used standards (grounding). A common middleware-to-middleware CommunicationAgent manages the coordinations among these proposed middleware sides.

4.6. Interactions Between DIANE Model Components and SDP

In this section, we present how SDP can exploit the DIANE components as SOA middleware for MANETs smoothly. Generally, SDP requires neither modifications

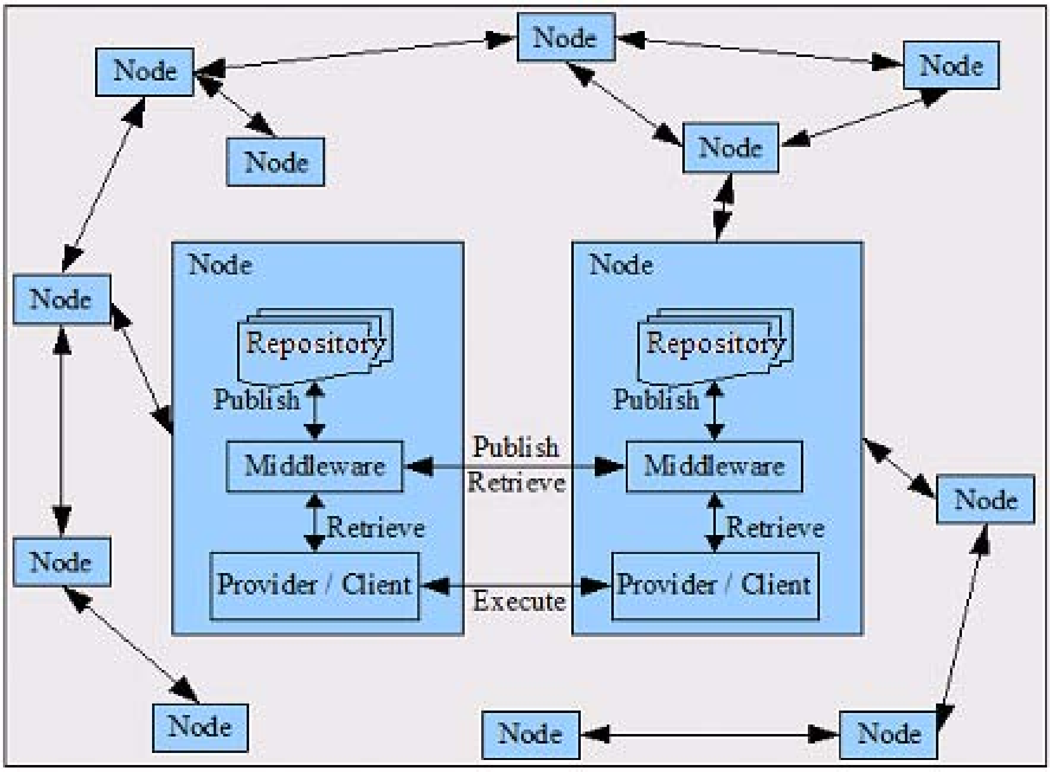


Figure 4.6.: DIANE SOA-Based Model

nor new components in the middleware in order to enable its replication approach.

4.6.1. DIANE Middleware and SDP Processing

Based on the detailed structure diagram for the DIANE middleware, presented in Figure 4.7, we now explain how SDP can be supported by the functionalities offered by the DIANE middleware in order to accomplish its service replication approach.

Service Usage

When a participant experiences an interest in some functionality, it expresses the required request and poses it to the centralizer component which forwards it to the communicator agent in order to be directed to the reachable repositories' agents. In the repository, suitable services and replicas are preselected. The highest ranked are selected and fed back to the caller agent of the requester via the communicator

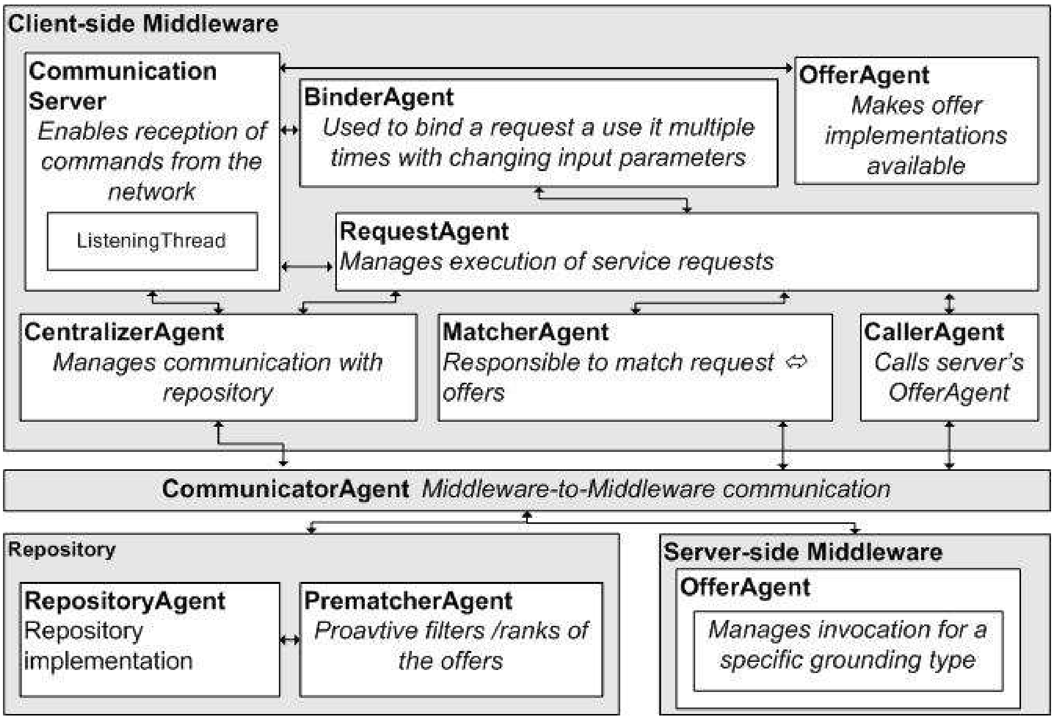


Figure 4.7.: Middleware of DIANE Model

agent. The caller agent starts invoking the service with the help of the communication agent and the service execution begins. The ranking returned by the repository can reflect both the service functionality and non-functional criteria, which will help to select the most interesting replica among a group of replicas.

Obtaining a Service Replica

SDP assumes that clients can count their requests to a specific service or any of its replicas. So, once a client achieved a certain replication threshold in terms of requests, it targets the replication process. The most interesting service or replica is selected from the repositories. Then, using the offered communication server, a replication process for the service and its resources starts between the current client and the service provider. Once a replica is received, the client-side middleware offer agent publishes the status of the new hosted replica at all the accessible repositories. As a provider, it should keep track that its offers are updated at all times in all repositories that it can access.

Activating a Replica

Once a service client achieves the replication threshold, it first communicates with its locally available repositories in order to search, if it has an inactive cached replica. If yes, it activates it using the next process of "Caching a Replica". The last step in this process is to publish the status of the newly changed replica at all the accessible repositories.

Caching a Replica

The second main SDP mechanism, namely hibernation, is presented by this process. Since SDP is an interest-based approach, each service provider counts its service interest, in terms of requests. Once the service interest is below a certain hibernation threshold, it hibernates the service by triggering the server-side middleware offer agent to update its offer in the accessible repositories to reflect the new inactive status.

The main intersections between SDP and DIANE SOA processes can be summarized as maintaining service repositories, discovery and execution processes.

4.6.2. Service Repositories

In a MANET environment, service repositories can never be realized in a centralized fashion as previously discussed. This distributed nature enables local partitions to have a certain knowledge about the global status of the available and hibernated services or replicas inside the network. This status reduction in the distributed repositories enables SDP to avoid analysis of the network statuses and topology. Instead of this, SDP utilizes the local repository knowledge with the failed requests to evaluate (either to replicate or hibernate) the available reachable services in its local network partition. SDP depends on the mobility of the nodes to let the service prevail in the other network partitions.

4.6.3. Service Discovery

SDP not only utilizes the service discovery and ranking modules in the DIANE framework but also addresses the challenge of using multi criteria, both functional and non functional, to rank the different, functionally equivalent replicas from a service interest point of view. More details are described about this in Chapter 6.

4.6.4. Service Execution

On the one hand, although SDP supports higher service availability by applying an innovative interest based replication approach, it does not force any style of service execution which should be considered as an advantageous feature for SDP. However, on the other hand, a tier for supporting the concurrent mobile service transaction management is still missing. Basically, SDP addresses a solution for service replication and answers the question of "when to replicate" easily without any required exhaustive processes of network analysis which requires expensive lower network layers' queries or special network architectures. Since in MANETs, no strict consistent transaction management can be ensured anyhow [DJ07, HM05], SDP considers a priority for the "when" question. Mainly, the service execution concepts may implicitly indicate a set of processes which includes service planning , invocation and serialization and messaging.

SDP and Service Planning and Composition

As depicted in the DIANE Middleware figure, the matcher agent is responsible for matching the request with the offers. The selected services are being composed -if required- to meet a set of requirements of certain offers if no single service can satisfy this request. Since SDP provides an approach which can ensure better distributions for popular service in MANETs, better service planning based on higher available services will be provided. In such an uncertain environment, the presence of the cached hibernated services is directly enriching the process of service planning and composition.

SDP and Service Invocation

SDP treats the replicas of the same service as individual services in a multimaster replication style [BDS07]. No seeking for the original service is required, as in [DB07]. So, consistent service invocation is presented by SDP. Moreover, from an invocation perspective, the required information for switching decision between services, i.e. switching for more interesting services, are also reduced in the repositories.

SDP Messaging and Stateful Services

The service target scope of SDP is wide. SDP is implemented in order to be decoupled of the applied messaging protocol, like SOAP [MNR05], and the related transportation protocols like HTTP. So, as an advantageous feature, SDP enables any messaging suite of protocols. Since SDP, as proposed, is based on a lazy replica consistency model, it assumes that the minimum time required for hibernation evaluation of two or more concurrently running replicas will be enough for applying a consistency schema to update active service states. As mentioned before, strict consistency can not be ensured in MANETs.

4.7. General Assumptions

In this section, most of the general assumptions required in the rest of the work are stated. Mainly based on the previous sections (4.5 - 4.6), SDP assumes the presence of the following components to be enabled:

- Service descriptions: In order to operate a service and the related functional and non-functional abilities and attributes, the service provider should have a formal service offer. The service offers can be described ontologically using some service Description language like the DIANE Service Description language (DSD) [KKRM05].

- Service repository: SDP needs service offers to be published in the service repository. As mentioned in the DIANE model as an example, the challenge of running in a MANET pushes SDP to maintain distributed repository. A managing component for keeping these distributed repositories updated and consistent is assumed to be found.

- Service discovery component: SDP assumes presence of an efficient service discovery mechanism. In Chapter 8, effects of specifying the time required by this component are presented. The required protocols for exchanging the offer specifications like Web Service Definition Language (WSDL) are assumed to be also found.

- Service ranking component: As mentioned in the basic concepts of SDP in Section 4.2, the alternative replicas of the same service are needed to be ranked. This rank is suggested to based on the service request cost or available resources at the provider. As we are going to show in Chapter 6, combining the assumed ranking decisions with the service popularity is addressed.

- Messaging component: SDP assumes that the provider and clients are using a suitable service messaging protocol like the Simple Object Access Protocol (SOAP) upon a compatible application layer transportation protocol like the Hyper Text Transfer Protocol (HTTP).

- Consistency and synchronization component: SDP is considering more to answer the main questions of when to generate and where to allocate a replica. Many mechanisms are contributed in the literature regarding the field of service consistency and synchronizations like [CPK+08, HM05, DJ07, MHKS08]. If there exists any needs, SDP considers a slot for a consistency and synchronizations components. It assumes anyway a presence of this component.

For the service to be replicated,

- Services are replicable. The formal definition of a replicable service is introduced in Chapter 5.
- All the required resources for a service to be replicated are available at the client side.
- Unless something else is stated, a service or one of its replicas can serve all of the clients in its network partition. The two last previously discussed assumptions are deprecated in Chapter 11.

4.8. Summary

In this chapter, we have introduced the main concepts of SDP. The proposed mechanisms of SDP have been evaluated on a scale of one network partition. The feasibility of applying SDP in a simple dynamic network partition (artificial partition) has been evaluated with promising results in terms of service availability and prevalence. Moreover, the issues of integrating the proposed basic SDP protocol in a candidate SOA model for MANETs is presented and discussed. Finally, the required components to be functioning to enable SDP in any SOA have been presented as assumptions. During the next chapters some of these assumptions will be deprecated and others will be realized and complemented with some SDP components.

Applying SDP in more realistic network model where there are many concurrently forming network partitions represents the main objective behind the work of the next chapter where extended descriptions and evaluations for SDP are introduced.

CHAPTER 5

Extended Descriptions and Evaluations of SDP

"The whole is more than the sum of its parts"

(Aristotle)

Based on the ideas and concepts discussed in Chapter 4, in this chapter, performance of SDP is described in an extended network model which describe a realistic MANET. The configurations for the applications of SDP in the modeled network are presented and discussed. Moreover, an extended performance simulation based analysis for SDP is introduced. The concepts of replica allocation correctness are introduced with a quantification method that can reflect how can SDP correctly allocate a replica relative to an optimal assumed service distribution situation. The work presented in this chapter has been partially published in [HKR08a] and [HKR09a].

The structure of this chapter is as follows: Section 5.1 presents an extended network model which describes a partitionable network with its resources. Moreover, formal definitions for the "replicable" services and the service replication process are proposed. In Section 5.2, a detailed extended evaluation based on simulation is presented. The results are investigated and analyzed. A presentation for another simulation based comparison study from the literature between SDP and DAR [AYS+09] is presented in Section 5.3. Section 5.4 summarizes the main advantages of SDP as a SOA and interest-based service replication approach. Finally, the work of this chapter is summarized, the contribution are highlighted, and the next research motivations are presented in Section 5.5.

5.1. Extended Network Model

The proposed extended network model considers the described model in Chapter 4. The resources of a MANET are distributed over the network participants and the wireless links which are formed during the mobility of the network nodes. Let us denote the whole resources of a given MANET at a certain time t as: $NET_{R|t}$, where $NET_{R|t} = NET_{LocalR|t} \bigcup NET_{LinksR|t}$, and $NET_{LocalR|t}$ is a set of the available resources offered by the mobile nodes locally at a certain time t, $NET_{LocalR|t} = \sum_{i \in NodeAddresses}^{NetworkSize} N_i(LocalR|t)$ and $NET_{LinksR|t}$ is the set of the available resources related to the currently formed wireless links like bandwidth,
$NET_{LinksR|t} = \sum_{i,j \in NodeAddresses}^{NetworkSize} E_{\{i,j\}|t}$.

5.1.1. Replicable Services

Based on the extended network model, a formal definition for the replicable service can be introduced. As mentioned in Chapter 3, services are not always replicable like data items. To propose a formal definition for the concept of a "replicable"

service, let S_x be a service X.
Let $R_{x|A,t}$ be the required resources for S_x to run at site A at a certain time t and $R_{x|A,t} = \{NET_{LocalR(A)|t} \cup RR_{X|t}\}$, where $NET_{LocalR(A)|t}$ is the set of the local resources at site A which are required by R_x at a certain time t and $RR_{X|t}$ is the set of the remote distributed required resources.
Let $(Permissions, RR_{X|t})_A$ be the set of permissions between S_x's host and the management systems of $RR_{X|t}$ from the service site A.
Let $CSN_{x|A}$ be the set of connected sessions to S_x where $CSN_{x|A} = \{session(S_x, Client_a)_t,\ session(S_x,\ Client_b)_t,\ \ldots,\ session(S_x,\ Client_n)_t\}$ and $\{a, b, \ldots,\ n\} \subset \{nodes'identifiers\}$ and $session(S_x, Client_n)_t$ is a connected session form client n to S_x at time t.
Let $dR_{x|t}$ be the required portion of resources by $session(S_x, Client_n)_t$.Let us define:

- $replicate(S_{x|A}, B)$ which is a set of functions and procedures that can produce $S_xReplica_B$ (a replicas of S_x at site A which is required to be allocated at site B).

- $replicate(dR_{x|A,t}, B)$ is a set of functions and procedures that can produce $dR_{x|B,t}$ which is a subset of the allocated required resources at B for replica B and a partial set of transfered resources form A to B.

- $replicate((Permissions, dRR_{X|t})_A, B)$ is a set of functions and procedures which can produces $(Permissions, dRR_{X|t})_B$ which is a subset of the required permissions list for a new $S_xReplica_B$ for its required remote resources $dRR_{X|t}$.

- $replicate(CSN_{x|A}, B)$ is a set of functions and procedures which can establish $CSN_{x|B}$ which is a subset of the active connected sessions, $CSN_{x|A}$, which were connected to S_x at A and migrated to be connected to $S_xReplica_B$.

Definition 1 ***: A replicable service***
S_x is a replicable service from site A to site B at time t if all of the following conditions can be realized:

- *$replicate(S_{x|A}, B)$ is feasible and $S_xReplica_B$ can be produced.*

- *$dR_{x|B,t}$ and $(Permissions, dRR_{X|t})_B$ are sufficient for $S_xReplica_B$ to run.*

- *the clients which are required to be migrated from site A to site B can find, discover and connect to $S_xReplica_B$.*

5.1.2. Service Replication

Based on the extended network model, the service replication process can be defined.

Definition 2 ***: Service replication as a process***
A process which can achieve each of replicate($S_{x|A}$, B), replicate($dR_{x|A,t}$, B), ($Permissions$, $dRR_{X|t}$)$_B$ and replicate($CSN_{x|A}$, B) when it is logically and physically feasible to produce and allocate a replica $S_x Replica_B$ at site B from a service S_x at site A (See Figure 3.2 in Chapter 3).

5.2. Extended Evaluation

As mentioned in the further research questions of Chapter 4, one of our main motivations is to evaluate the proposed replication protocol, SDP, in more realistic network models. The extended network model in Section 5.1 introduces a MANET with a partitioning behavior instead of the concept of artificial partition in Chapter 4. Principally, we re-investigate the performance of SDP but with the proposed extended network model. Moreover, we extend the evaluations to include an analysis for the optimality of the generated service distributions and the replica allocation process of SDP over the multiple network partitions. Therefore, A detailed simulation for SDP has been performed (based on the simulator described in Appendix A). Results stem mainly from two groups of settings: In the first group we apply just the replication mechanism (R group), and in the other one both replication and hibernation mechanisms(R-H group) are applied together. The R group represents the worst (most costly) solution for an interest based replication approach, in which each replica will remain active forever after being hosted by any of the mobile nodes. The R-H group represents the real expected performance of SDP since both replication and hibernation mechanisms are activated. In our performance analysis, four main performance metrics have been analyzed. The same performance metrics which are defined in Chapter 4 are used.

- *Service Availability*
- *Service Prevalence*
- *residence time*

Beside using these metrics, we use the *Success ratio.* The success ratio is the ratio of the number of successful service requests to the overall number of requests in the entire network.

5.2.1. Service Model

The network maintains only one service at the beginning of its operation time. This service is placed on the first created node in the network. Three assumptions are made in the service model: (a) all mobile nodes can participate in the replication mechanism, (b) the original service is replicable and (c) all participants can cache and restore the replicas in case of service hibernation. Each replica's offer is labeled by a requirement index which is normally distributed of 20% about a general requirement index, see Chapter 6. Clients search for the minimum requirement index from the neighboring services to pose their service requests to.

5.2.2. Requesting Behavior Model

The requesting model indicates the requesting behavior of a client regarding a specific service at a time. Initially, all nodes seek for the initial (original) service provider node. Only those nodes with a feasible path to the provider node will be able to start evaluating the service requesting and be involved in the related replication/hibernation processes. After a while, service/replicas prevalence through the network is supposed to cover as much as possible of the ad hoc formed partitions. Variant requesting rates are maintained by each node; the requesting rate is generated between [0..3] requests per minute, the requesting rate is constant during a requesting period of {10,20,...,50} minutes, and after a pause time of {0,3,6,...,12} minutes, the node selects another requesting rate and so on. Requesting rate, requesting period, and pause period are uniformly randomly generated. More details about different requesting behaviors and how to quantify these behaviors are mentioned in Chapter 6.

Table 5.1 shows the rest of the applied configurations and settings in the current simulation.

In case of varying the network size, the mobility index is fixed to be 50% and the maximum allowed prevalence is set to be 100%. In case of observing the effects of varying the mobility index, the network size is set to be 50 nodes, and the maximum allowed prevalence is set to be 100%. Finally, in case of varying the maximum allowed service prevalence, the network size is fixed to be 50 nodes, and the mobility index is 50%.

5.2.3. Performance Analysis

Service Availability:

In Figure 5.1, regarding the R group curves against network size, mobility index and maximum prevalence, the service availability ratio is always 1, and this is logical because no service and no replicas could ever be turned off (hibernated), so

Configuration parameter	Value in simulation
Number of nodes	10 to 100 nodes
Field size	$600m \times 600m$
Transmission range	75 m
Mobility Model	Random waypoint [LNR04]
	$Speed \in [0, MaxV]$
	$MaxV \in [0, 12 \times MobilityIndex/100]$ m/s
	$PauseTime \in [0, MaxT]$
	$MaxT \in [0, 30 \times (1.0 - MobilityIndex/100)]$ minutes
Simulation time	7200 seconds
Number of runs	20 runs
Replication threshold	3requests/1minute
Hibernation threshold	1request/3minutes

Table 5.1.: Configurations of the extended simulation and analysis of SDP

they remain running till the end of the simulation time. On the other hand, in the R-H group, the service availability is growing as the network size increases. The smaller network sizes can not ensure enough requests that save the service and its replicas from being hibernated. Moreover, regarding the partitioning behavior of the network topology, even if there is interest by some clients in the network for the service, there is no guarantee that their requests will reach any provider node (regarding the fact that they are in different network partitions). This effect disappears as the network density increases and lower number of network partitions are formed. The higher sizes also ensure higher number of requests which ensure longer life time for the replicas. Starting form 70 nodes, there are about 11.3 partitions with standard deviation about 1.6 partitions and the service availability is more than 0.85. Generally, SDP achieves in average about 0.78 for service availability with about 0.27 related standard deviation in cases of varying sizes of the network. Regarding varying the mobility index, for the R-H group, the service availability is about 0.92 with a standard deviation of 0.12. As shown, as the mobility increases, the service availability increases,too. This is logical because as the mobile nodes traverse the different network partitions, replicas gain more requests which positively affect the service availability.
No clear pattern could be interpreted from varying the maximum allowed prevalence. Service availability is about 0.92 with a standard deviation of 0.22. This result is due to the fact that SDP requires a low service prevalence, as shown in Figure 5.3.

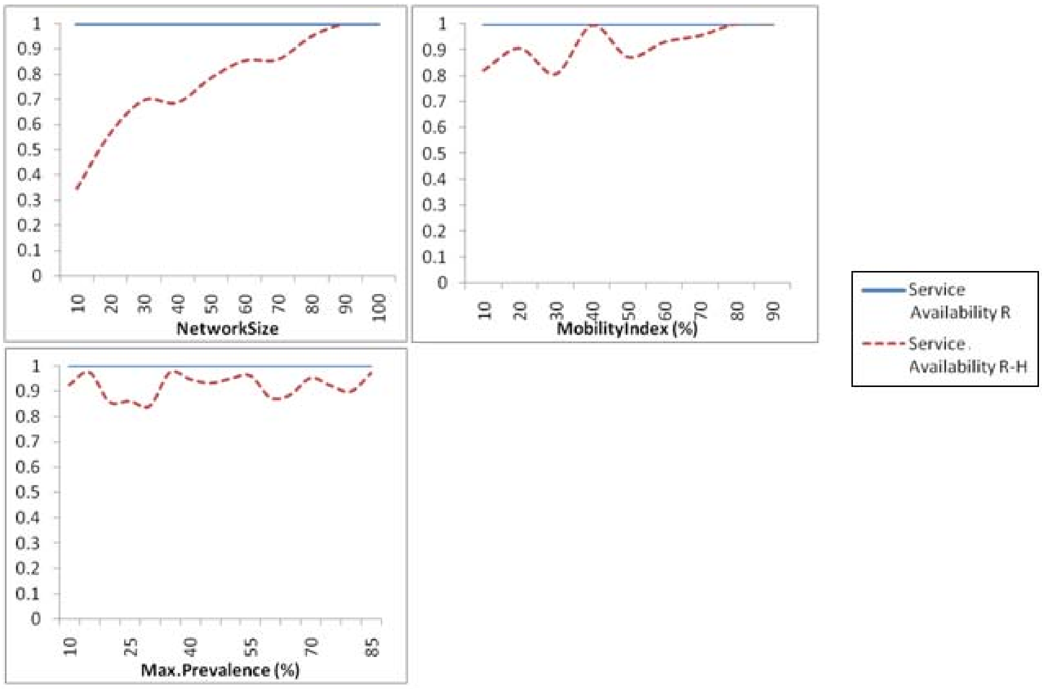


Figure 5.1.: Service availability

Success Ratio:

In Figure 5.2, as unlike the service availability, the success ratio is not granted for the R Group, it increases as the network size increases. In case of higher network sizes, lower numbers of network partitions, the success ratio becomes higher because in more dense networks, paths to service providers are more available. Starting from 70 nodes, the success ratio is about 0.90 with a standard deviation of 0.03. The average value of the success ratio for the R group regarding varying sizes of the network is 0.73 with a standard deviation of 0.09. The same behavior is shown for the R-H group. The success ratio increases as the network size increases. The average value of the success ratio for R-H group regarding varying sizes of the network is 0.57 with standard deviation 0.18. Starting from 70 nodes, the success ratio is about 0.75.
With respect to varying the mobility index, for both the R and R-H group, the success ratio increases as the mobility increases. Starting from 50% mobility index, it is about 0.92 with a standard deviation of 0.02 for the R group and about 0.73 with a standard deviation of 0.3 for the R-H group. Generally, SDP achieves a success ratio of about 0.73 with a 0.18 standard deviation for the R-H group for all mobility indices.

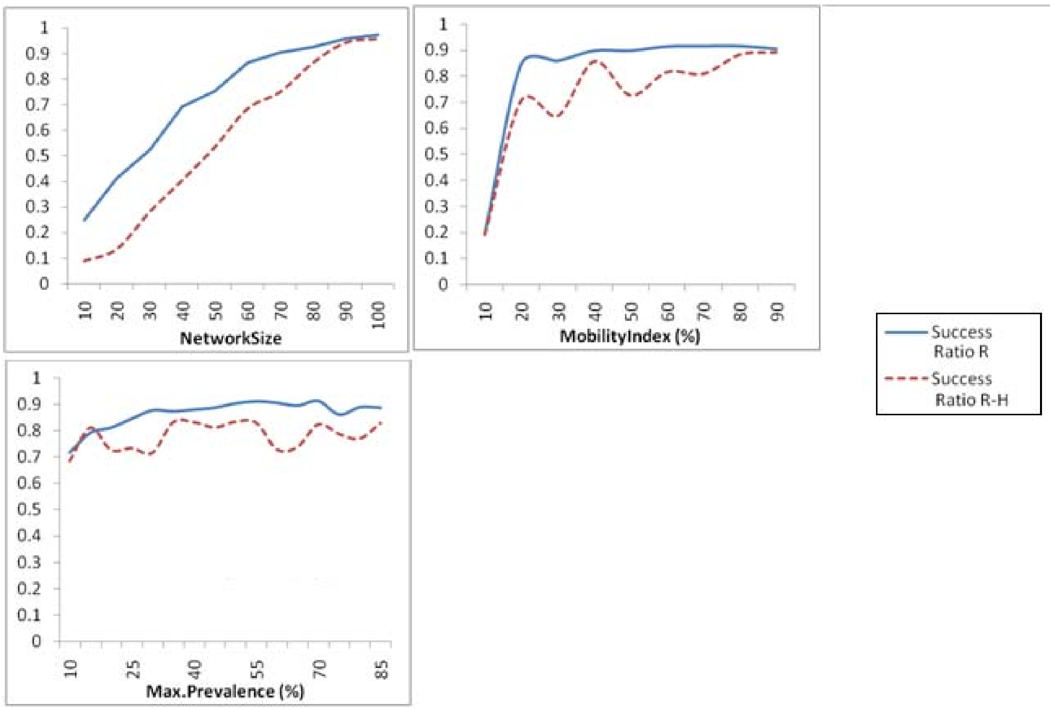


Figure 5.2.: Success ratio

Lower values of the maximum prevalence tune down the success ratio, but, as we are going to show later in Figure 5.3, since SDP has a low prevalence ratio, no clear effect of allowing higher values of maximum prevalence appear. The general average success ratio for the R group is 0.85 with 0.04 standard deviation, while it is 0.78 with standard deviation 0.21 for the R-H group.

Service prevalence:

Figure 5.3 shows a high difference between the service prevalence resulting from both the R and R-H groups against the network size, mobility index, and the maximum allowed prevalence. While the prevalence in R-H group is , on average, less than 0.15 with standard deviation 0.07 for all network sizes, it grows to be about 0.65 with standard deviation 0.07 at 100 nodes.
Regarding the mobility, the service prevalence increases as the mobility index increases. For R group, prevalence ratio continues increasing to be about 0.60 starting from 20% mobility index , while it becomes less than 0.19 with standard deviation 0.05 starting from 20% mobility index. The same behavior could be seen of varying the maximum allowed prevalence for both of R and R-H groups. The high differ-

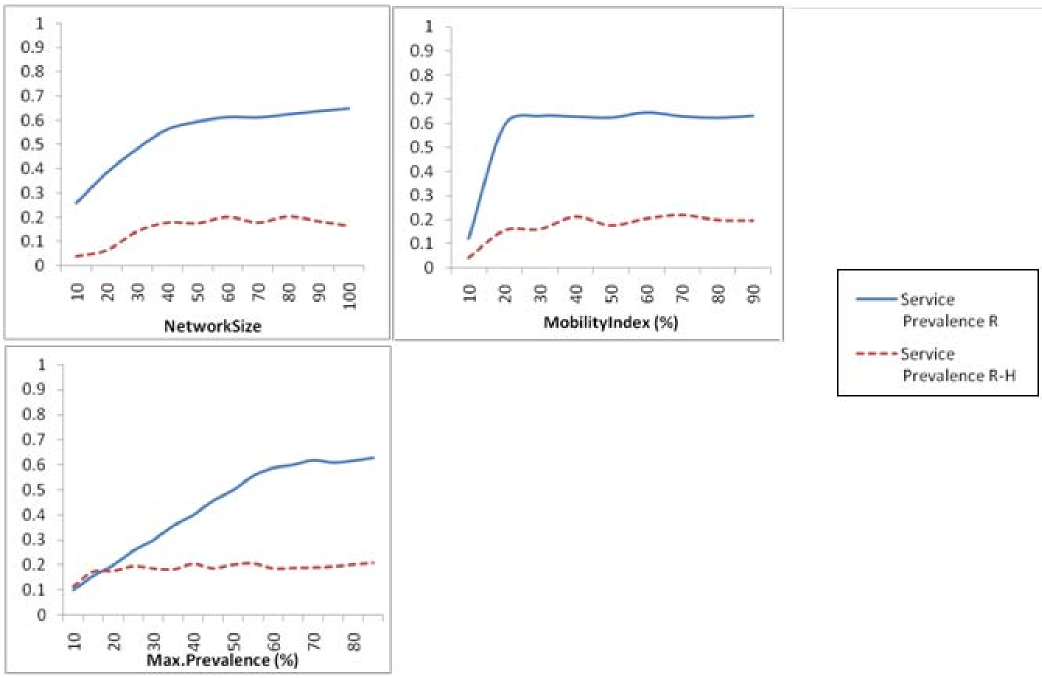


Figure 5.3.: Service prevalence

ence between the resultant prevalence of R and R-H groups indicates the effect of the hibernation mechanism. The prevalence reduction of the number of the running replicas minimizes the required effort of replicas synchronization and of course network resources' utilization.

Residence Time:

Regarding varying the network sizes, Figure 5.4 shows that applying only the replication mechanism in the R group produces more or less constant residence time about 01:14:15 (hh:mm:ss) with 00:07:32 (hh:mm:ss) standard deviation starting from 20 nodes. On the other hand, the residence time decreases as the network sizes increases for the R-H group with general average 00:13:15 with 00:07:28 standard deviation. The huge reduction could be also noticed when varying both mobility index and maximum allowed prevalence. This reduction can be explained from the fact that when the set of the hosting nodes are supposed to be increased, not all participants can receive the same client interest portion so many providers trigger the hibernation mechanism and shut down their service. This behavior makes the average residence time decrease affected by both increasing the mobility index and the maximum allowed prevalence.

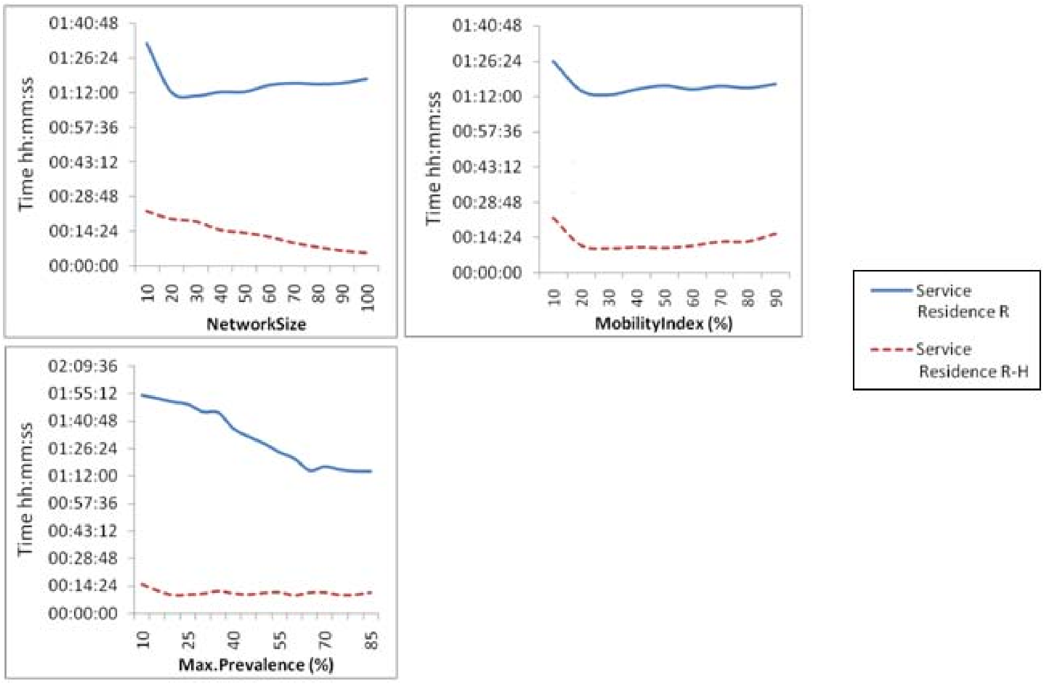


Figure 5.4.: Service residence time

5.2.4. Replica Allocation Correctness Analysis

In this subsection, we extend our analysis of the simulation results, the analysis is done on results of applying both replication and hibernation mechanisms together (R-H group) only.

Replica Allocation Process:

One of the main question to be taken into consideration by SDP is how to place a replica in the network is a . Obviously, if we can put at least one replica inside each formed network partition, it will be the optimum solution (service distribution) for the replica allocation process (if one replica can satisfy all the generated requests inside its network partition). Unfortunately, this optimum distribution can not be guaranteed by our approach. We claim that it will also be very hard to be achieved by any other replication approach for MANETs. Our proposed measure of the correctness (optimality) of the replica allocation (placement) process will be the Correctness ratio which is a relation between the number of available active replicas inside a given partition and its size. For simplicity, one active replica is assumed to be enough to satisfy all requests of one network partition. This assumption is deprecated in Chapter 11. Two correctness ratio computation methods are proposed in this chapter namely linear and rational correctness ratios.

Linear Correctness Ratio: The Linear Correctness Ratio (LCR) of the allocation process is bounded between 0% and 100%. If there is no replica in the partition the ratio should be 0%. Else, if there are one or two replicas in the partition, the ratio will be 100%. Otherwise, for simplicity, the ratio is linearly inversely proportional to the number of replicas. The ratio becomes zero at a number of replicas equal to the partition size (Pz). Normally, at least one replica per partition is an optimal case. Finding two replicas in the same partition is very healthy from a replication point of view. It is a required behavior, since we employ a replication approach. At least by the end of a replication process for once, there will be two replicas inside the same partition. Therefore, we do not penalize the replication behavior for two replicas. The following equation describes the value of the Linear Correctness Ratio $LCR_t(P_i)$ in an ad hoc formed partition P_i at a certain moment in time t:

$$LCR_t(P_i) = \begin{cases} 0 & R_i = 0 \\ 1 & R_i \in \{1, 2\} \\ \frac{Pz_i - R_i}{Pz_i - 2} & R_i > 2 \end{cases} \tag{5.1}$$

where R_i: the available number of replicas in the i^{th} partition.

Rational Correctness Ratio: Rational Correctness Ratio (RCR) of the allocation process is also bounded between 0% and 100%, but it is a rational basis relation to be more sensitive to higher numbers of active replicas inside the partition it can be computed as follows

$$RCR_t(P_i) = \begin{cases} 0 & R_i = 0 \\ 1 & R_i \in \{1, 2\} \\ \frac{3}{R_i} \cdot \frac{Pz_i - R_i}{Pz_i - 2} & R_i > 2 \end{cases} \tag{5.2}$$

In Figure 5.5, while LCR increases with higher network sizes, RCR stops increasing at a specific network size (about 60 nodes) and starts decreasing. This behavior can be interpreted from a perspective of the number of formed partitions . The number of formed partitions increases as the network size increases up to 60 nodes (about 11 partitions) the number of formed partition reflects the behavior and starts decreasing as the network size increases, this makes the produced replicas be hosted mostly in the same partition. Especially in dense networks, this means presence of higher ratios of LCR does not reflect the real replica distribution situation which is presented by RCR.
As the mobility index increases both LCR and RCR increase too. This is due to

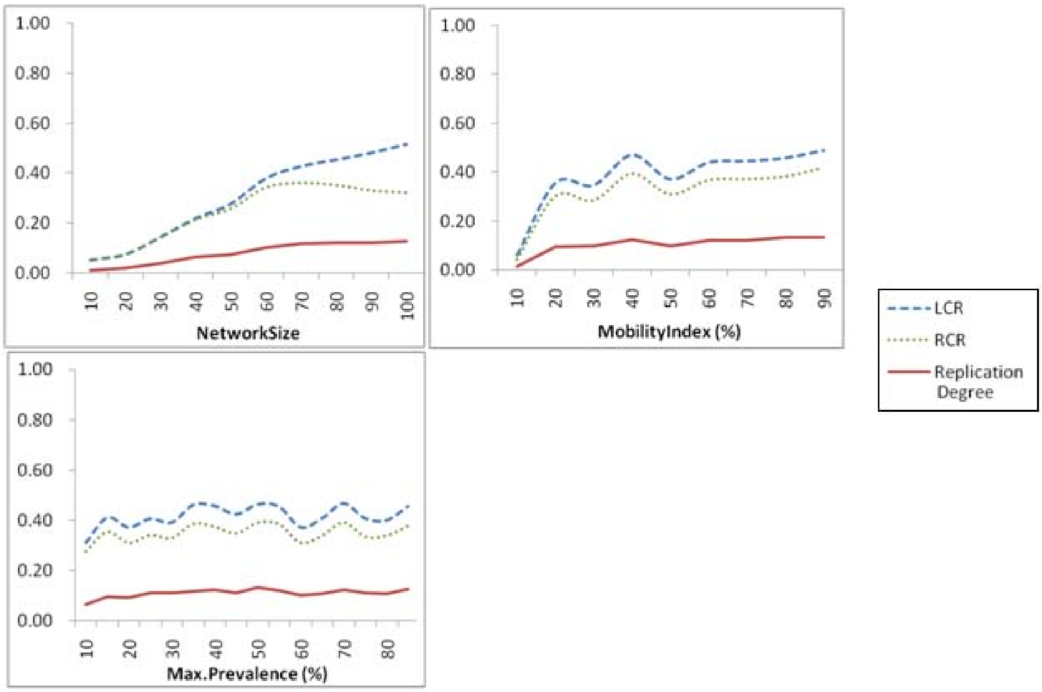


Figure 5.5.: Replication degree and allocation correctness: Linear Correctness Ratio (LCR) and Rational Correctness Ratio(RCR)

the fact that more mobile nodes in the network will allow the service to prevail through the different network partitions.

Replication Degree:

Concepts of service prevalence, replication degree, and replica allocation correctness ratio are closely related and could be interpreted as different terms for indicating the number of running replicas inside the network. Each of these parameters has its own perspective in highlighting the number of concurrently running replicas: while the service prevalence measures that number on the scale of the whole network, the replication degree summarize the status of the service distribution in the network on the level of the network partition, and the computation methods of the replica allocation correctness represent a judgement for how each of the network partitions is close to the optimum distribution of replicas in a given network. Figure 5.5 shows that SDP achieves a very low replication degree which seems to be varying about some average about 0.10 against varying network size, mobility index, and maximum allowed prevalence. As replication degree refers to the replication cost, results are very promising from the perspective of utilization of network resources.

Further motivation questions:

The first part of this chapter shows that it is feasible to apply SDP in a more realistic MANET model (see Section 5.1). It provides answers to the motivation questions of Chapter 4. At this stage of our work, we have the following questions to be investigated:

1. What are the effects of the requesting behavior of the client on the service distribution process? With which requesting behavior specification can SDP operate better? What the meaning behind these specifications? How can we model service popularity based on the requesting behaviors?

2. What are the effects of different replication and hibernation behaviors on the performance of SDP?

3. What are the effects of different mobility settings and models on the performance of SDP?

4. What should the performance of SDP be relatively to the other approaches?

The first set of questions presents the motivations for Chapter 6. Chapters 7 and 8 deal with different replication and hibernation behavior to investigate their effects on the service distribution process of SDP. The last two motivation questions are investigated in Chapter 9.
Furthermore in the next section, we present a comparison in performance between SDP and other service replication approach for MANETs. This comparison can illustrate the SDP performance relatively to other replication alternative approach. Although this performance comparison investigates partially the performance of less-configured SDP, it gives a positive feedback and supports our thoughts about SDP concepts. Keep in mind that in Chapter 9, more detailed comparisons between SDP and other two replication approaches are presented where SDP considers the rest of the previously mentioned motivation questions.

Configuration parameter	Value in simulation
Number of nodes	25 to 200 nodes
Field size	$500m \times 500m$
Transmission range	100 m
Bandwidth	2 Mbps wireless channel
Routing protocol	Ad Hoc On Demand Distance Vector (AODV)[PBRD03]
MAC protocol	IEEE802.11b without power control
Mobility Model	Random waypoint [LNR04] with no pause intervals
Simulation time	400 seconds

Table 5.2.: Configurations of DAR comparison experiments to SDP [AYS$^+$09]

5.3. SDP Compared to DAR

As previously discussed in Chapter 3, DAR is an energy-aware service replication protocol for MANETs. DAR requires to know information about the current neighborhood of each mobile node. So, it is also topology-aware replication approach. In [AYS$^+$09], a comparison has been made between the SDP performance to DAR. The authors of DAR, using the QUALNET simulator[1], elaborated a set of experiments. The specifications of these experiments are shown in Table 5.2. Service availability, prevalence and energy consumption have been evaluated versus both the maximum allowed speed in (meters/second) and the network size. A minor modification has been introduced on the service availability. The average service availability in [AYS$^+$09] is computed as a function of the hop counts (between both client and provider) and the network size. The authors assume that the shorter the distances between the client and the server are, the higher service availability is. Therefore, form a client perspective, the *Service availability* $= \frac{1}{HopCount(c_i,S)}$ of the i^{th} client c and service S. In general, the *Average service availability* $= \frac{1}{n} \times \sum_{i}^{n} \frac{1}{HopCount(c_i, S)}$, where n is the number of S clients.

Regarding the service requesting specifications, each of the clients generates a request rate under the Poisson distribution with average λ equals 6 requests (each is 512 bytes) in time interval of 60 seconds. Initialy, there is only one service in the environment with size of 2K bytes. Regarding the used specifications for both replication and hibernation thresholds, [AYS$^+$09] suggested four cases by using several different replication and hibernation thresholds. The first and third cases are denoted (SDP-R 4) and (SDP-R 5) respectively where the replication mechanism is only applied with 4 (for the first case) and 5 (for the third case) requests per 50

[1]QualNet simulator home page: http://www.scalable-networks.com/

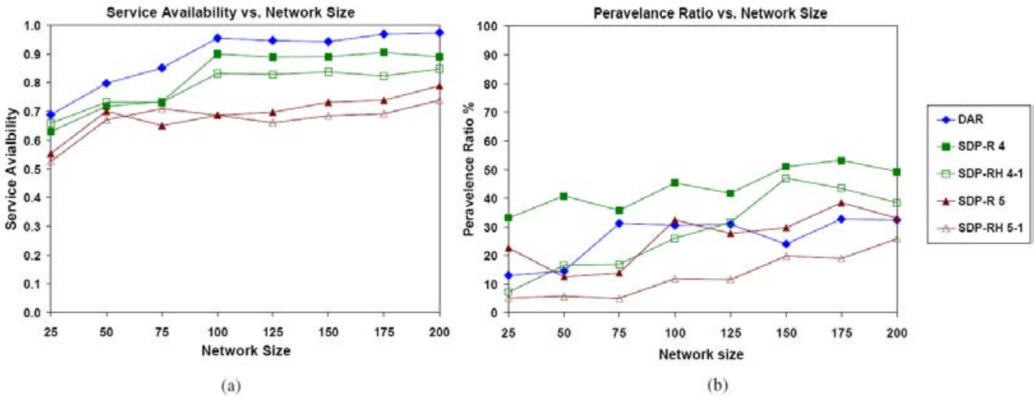


Figure 5.6.: Service availability and prevalence vs. network size [AYS+09]

seconds as replication thresholds. In the second and fourth cases, both replication and hibernation mechanisms are applied. Both cases are denoted (SDP-RH 4-1) and (SDP-RH 5-1) respectively where the hibernation threshold is 1 request per 50 seconds.
The independent variables in the presented simulation study of [AYS+09] are the network size (between 25 and 200 nodes) and the maximum allowed node speed (1, 2, 3, . . . , 10 meters/seconds). The minimum speed is (0 meters/second). When the network size is varying, the maximum allowed speed is fixed at 1 meter/second and when the maximum allowed speed is varying, the network size is fixed at 100 nodes.

5.3.1. Service Availability and Prevalence

As shown in Figure 5.6, regarding the defined zone-based availability, DAR can achieve higher availability tan any of the SDP replication hibernation specifications. On the other hand, DAR achieves relatively similar prevalence ratios compared to the different SDP specifications. SDP can achieve (with some specifications) quite close service availability and usually lower (better) prevalence ratios.

5.3.2. Energy consumption

Motives behind DAR and SDP protocols are deferent. While DAR aims to reduce the overall energy consumption by the service replication protocol by querying the routing component and logically dividing the network into smaller zones, SDP aims to avoid the complications of inquiring the lower network layers such as the routing component or performing any topological analysis. Therefore, at this stage of our work, we are satisfied by this comparison between SDP and DAR. Moreover, we

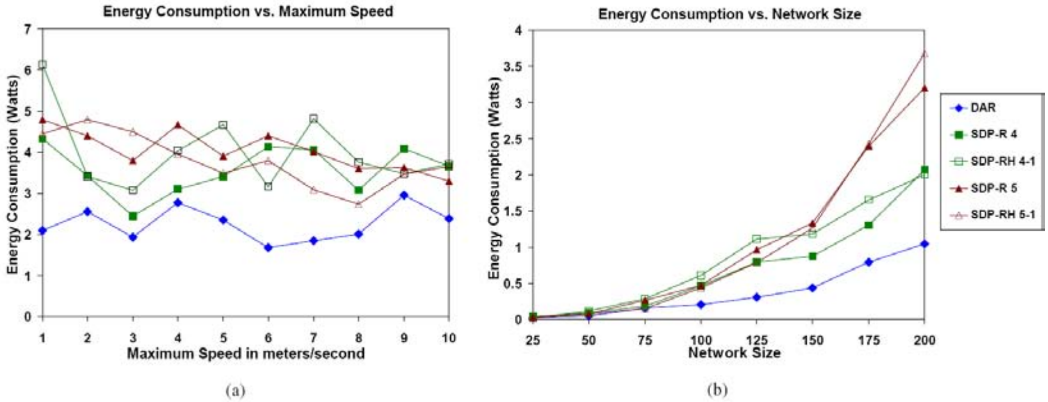


Figure 5.7.: (a) Energy consumption vs. maximum speed and (b) Energy consumption vs. network size [AYS+09]

think that the four suggested cases by DAR authors for SDP are very close to the energy consumption of DAR, especially for the high speeds and the moderate to low network size as shown in Figure 5.7.

Comments on the Comparisons to DAR: SDP specifications of the replication and hibernation thresholds represent the clue for better SDP performance. As previously mentioned, sometimes the performance of SDP is close to be better than DAR. This good performance is due to the specifications of both replication and hibernation thresholds. Moreover, the mentioned comparisons stated nothing about the correctness of the DAR replica allocation correctness. The clue is in better SDP specifications. We think that it is a problem of specifying better replication, hibernation thresholds and more meaningful requesting behaviors of the clients. During the next chapters of this thesis, enhancing the performance of SDP is an objective. Finding a proper set of specifications for these specifications is going to be investigated. Addressing the issues related to these specifications is introduced in Chapters 6, 7, 8 and 10.

5.4. SDP Advantages

After realizing the basic SDP concepts of Chapter 4 and the extended application and analysis on a realistic network model in Chapter 5, the advantages of SDP features of the SDP are presented in this section. Saving the resources in a MANET is the clear advantage of using SDP for increasing service availability and enhancing service execution. Mostly, the advantage of SDP result from being based on a SOA architecture. These advantages can be summarized as follows:

1. Network architecture independence: For MANETs' service replication protocols, excluding SDP, there exists a strong relation between the protocol functionalities and the suggested used network components, like routing algorithms [S. 99]. Moreover some of these protocols recommend using specially equipped mobile nodes, for example with GPS, in order to complete the topological analysis processes [JJKY04]. This relation drives a coupling between the replication protocol and the network architecture required to be used. The main reason behind this coupling in these protocols is to cover projectively, using any prediction scheme, the formed partition by all of the available services inside the parent partition. Instead of that, SDP is using a combination of interest and the reduced information "descriptions" about the services in the local repositories to allow the service prevalence in all partitions. No more partition analysis and predictions are required by SDP.

2. Service content volume: This is an important term for SDP. It refers to the set of available services for the replication process (the service collection). Since SDP is using the basic SOA popularity-based service selection mechanism, it offers categorization for the service volume. Not all of this volume should be replicated. Only the interesting services are supposed to be replicated, others should not. In contrast to SDP, other proposed protocols have no direct guidance in this issue which means that they will pay the same replication effort for all of the service volume irrespective of the importance of its contents. More details about the role of the service content are addressed in Chapter 6.

3. Variant replication effort: Based on the popularity, not all of the services will get the same replication effort and the related allocation of resources by SDP.

4. Self adapting approach: Continuous replication and hibernation processes which are based on a varying service interest ensure that the replica distribution in the network is usually converging to the optimum service distribution. A set of suitable settings are determined in Chapters 7, 8 and 10 to enable SDP to adapt the service distribution in the network. Chapter 8 states the importance of determining the minimum required time for publishing and requesting (discovery and invocation for an offer and a service) in the process of leader election and how it can ensure a service distribution which tenses usually the optimum service distribution.

5.5. Summary

In this chapter, an extended description for the core mechanisms of SDP has been presented. A set of performance metrics for SDP in an extended network model has been addressed. Moreover, the correctness issue of the replica allocation process has been introduced. Two basic allocation correctness ratios have been proposed. The correctness computation methods can indicate how correct a service distribution situation is relatively to an optimum assumed service distribution. SDP showed good and promising results in terms of the proposed performance metrics and the correctness ratios. In the second part of this chapter, we have given a digest of a citation from the state of the art which compares SDP to another alternative. This comparison showed that SDP performance is comparable to that of the other approaches. At this stage of our work, this is a satisfying result. Finally, after gaining the promising results of the extended analysis of SDP, we stated the advantages of SDP as a SOA and interest-based service replication protocol.
Since SDP is an interest based SOA service distribution approach, service interest concepts need to be more polished. Although service popularity is very important, it is not well described or investigated with its influence parameters. These parameters can also be interpreted on many levels such as functional, non functional and QoS levels. The request behavior of a group of clients regarding a specified replica generates a common interest for this service which is the *Gross Interest*. The service replication/hibernation processes are dependent on the gross interest. Based on the clients' request behavior, the gross interest can be described. As mentioned before in Chapter 4, SDP requires to index the available equivalent services and replicas inside the available service repositories. Finding a general index to be used here is not easy.

Before addressing the issues of finding a proper set of specifications for both the replication and hibernation behavior of SDP, understanding the service popularity and its dimensions in our research has a higher priority. Thus, the following chapter considers more details about the service popularity and the gross interest as a popularity indication. Moreover, more details about the proposed service "requiremets' index" will be given. Then, the effects of different specifications for both replication and hibernation behaviors of SDP will be investigated in Chapter 7.

CHAPTER 6

The Gross Interest: Service Popularity Aggregation

"If I have a thousand ideas and only one turns out to be good, I am satisfied"

(Alfred Nobel)

Service popularity, e.g., how often a service is requested, can be an important non-functional property determining the life-cycle of a service. To capture it, the requesting behavior of clients needs to be modeled. In this chapter, we introduce and discuss: the importance of the service popularity, a generalized requesting model that can capture the requesting behavior of clients, a service popularity measure called “Gross Interest”, and a Gross Interest quantification method. Two extremely different sets of specifications for the proposed generalized requesting model which produce two different Gross Interest scenarios (rich and poor scenarios) are introduced and quantified. The work of this chapter has been partially published in [HKR11], [HKR08c], [HKR08a], [HKR09a], [HKR08b] and [HKR09c].

The structure of this chapter is as follows: Section 6.1 presents the importance of the service popularity as a non-functional attribute of services. A generalized service requesting behavior model for a client is proposed with its criteria in Section 6.2. Moreover, the Gross Interest as a quantification for the service popularity is introduced. In Section 6.3, different specifications for the requesting behavior scenarios are proposed and analyzed. The effects of the service popularity and the Gross Interest on enhancing the service contents of a network are addressed and discussed in Section 6.4. Finally, the work of this chapter is summarized, the contribution are highlighted, and the next research motivations are presented in Section 6.5.

6.1. Introduction

Selecting a suitable service or a set of services is usually based on the service ranking process. Service ranking is an important process which utilizes the service descriptions to determine an index of relevancy for a set of services regarding a service request. The ranking process generally uses two main categories of service description attributes which are *functional* and *non-functional*. While the functional and behavioral attributes describe respectively what the service does and how its functionalities are achieved, the non-functional attributes describe the restrictions and constraints on the provided service functionalities [TRF$^+$07]. Matching a service offer to a specific service request is dependent on the deployed service ranking and matchmaking process. The early service ranking and matchmaking approaches focused only on the functional and behavioral attributes. Recently, a new generation of ranking approaches life time is starting to include the non-functional attributes in their decisions. Finding a way to balance the impacts of the functional and non-functional attributes is the key for achieving a better service ranking and more accurate matches [HKRK07].
Even if the descriptions of a specified service include the functional and typical non-functional attributes, they miss an important category of attributes, namely those

aggregated attributes that the service can accumulate during its life time. This lack in descriptions can not be complemented or configured by an initial service offer. Attributes like service popularity and trust represent an important set of attributes that can not originally (or at any other time) be specified by the service provider. Instead of that, it is accumulated, for example, from its clients requesting behavior. From a provider perspective, the service popularity plays a great role. It is needed for a service provider in order to decide when it is supposed to enhance, replace or shut down a specified service. From a client perspective, service popularity plays for example an important role in supporting the shown trust attributes of the service provider. Moreover, considering a set of competing services and providers, having such computed service popularity attributes supports deploying better versions of services.

The concepts of service ranking and popularity may seem overlapping, but, imagine a distributed system with a SOA enabled application in which an un-interesting (i.e., unpopular) service is deployed with a unique functionality. Should there ever be any request for this functionality by a system participant, this service will perfectly match, but this happens rarely. On the other hand, this service needs to be considered by the SOA core mechanisms like discovery and matchmaking adding to the effort required to perform them. Moreover, in some environments like unstable and mobile networks, this will be even more extreme, as there, services will frequently need to be replicated to ensure availability as required in many cases and applications. Replicating a service that is basically never used is clearly a waste of resources. From another perspective, both service ranking and popularity may be overlapping, if the service popularity reflects better functional, behavioral and non-functional service facilities. In that case, the better service offers will be more popular. In these cases, the service popularity that is being computed during the service operation will be not only affected by the good attributes of the offered service, but will also prove these offers.

Since aggregated attributes, namely service popularity, are absent in the service offer, the current service ranking and matchmaking approaches can not include them in the service selection. So, this missing integration between the service popularity and selection is required to be overcome.

In order to have a measure for the service popularity, many candidate attributes can be considered. Examples are: the number of requests that a service gains in a certain period of time, the trust values set by different clients, the vitality (such as DHCP services) for the system participants and the service provider's stability and available resources. These can form individually or in combination the main measurements behind the service popularity.

In this chapter, the concept of *Gross Interest* as a measure for the important non-functional property "service popularity" is introduced. Since, the *Gross Interest* is a non-functional measure based on the clients' requesting behavior which aggregates popularity of a certain service at a specific point in time, a general service requesting model is described and presented based on the request behavior modeling of Chapters 4 and 5. The properties and abilities of this requesting model are described and analyzed.

6.2. Service Requesting Behavior

Based on the offered functionality of a specified service and how important this functionality is for all/some of the network participants (clients), a client forms his interest in this service. The client service interest is reflected in the generated service requesting behavior. From a service provider perspective, aggregating the different clients' interest regarding the offered service in one measurement is vital. The term *Gross Interest* refers basically to the aggregated interest towards a specified service within certain time.
Frequency of requests, amount of transported data and workload are the of main factors that affect the service Gross Interest. Based on one (or combination) of the mentioned factors, a service provider can estimate the popularity of his offered services. Moreover, in case of presence of services with the same functionalities, the services can be ranked based on their interest or popularity. For the required popularity ranking process and in order to hold a common ranking for the different(same functionality) services, a reference value for the highest (possible to be achieved) Gross Interest should be determined and taken into consideration. In this section, a set of considerations for how to quantify the Gross Interest based on the number and frequency of the generated requests regarding a specified service is given. In order to quantify the Gross Interest, a generalized model that can capture any client requesting behavior regarding a specified service is introduced. Then, the important criteria of this requesting model are presented and discussed.

6.2.1. Modeling The Service Requesting Behavior

[HKR08c, HKR08a, HKR09a] introduce a basic "calling" model which can describe the behavior of the generated requests regarding a specific service by a client. Based on this calling model, we propose here a generalized requesting model is proposed. The generalized model can capture a wide set of different requesting behaviors by clients for many services. As depicted in Figure 6.1-A and B, any client requesting behavior regarding the offered service can be modeled as follows:

- As in Figure 6.1-A, the operation time of the service can be divided into sequence requesting cycles from the perspective of clients.

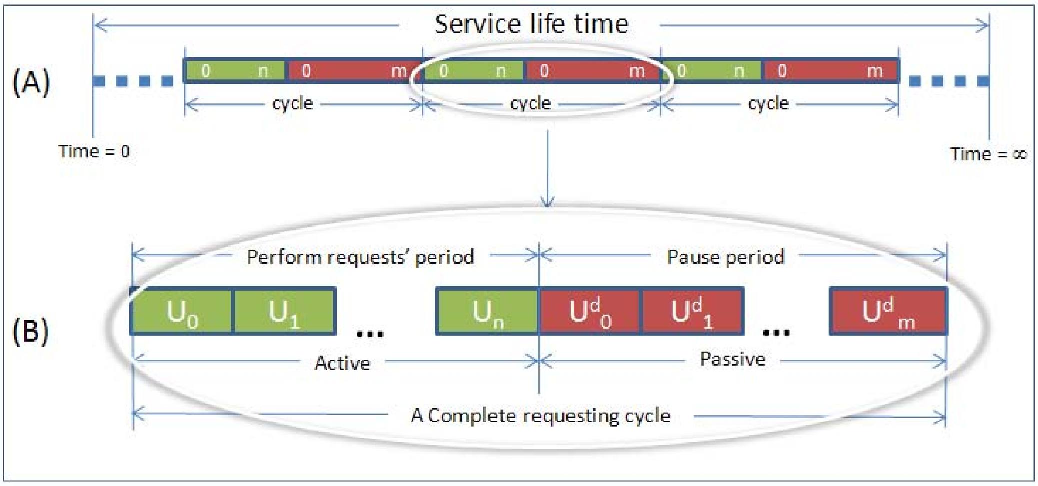


Figure 6.1.: Service requesting cycles from a client perspective

- As in Figure 6.1-B, each of the requesting cycles consists of two periods: an "active" period in which the client performs requests to the service, and a " passive" period where the client pauses generating requests to the service.
- The active periods are composed of a number of units Us with a predefined length in time. The length and the number of these units N varies from cycle to cycle and depends on a given distribution for each requesting cycle.
- During the active periods, a requesting rate, $RequestRate$, describes the frequency of generating the requests. The value for the requesting rate varies from cycle to cycle and depends on a given distribution.
- The passive periods are composed of a number of units U^ds with a predefined length in time. The length and the number of these units M is varying from cycle to cycle and dependent on a given distribution for each requesting cycle.
- The model specifications (U, U^d, N, M and $RequestRate$) for each requesting cycle are allowed to vary. Each of the clients may have its own specifications of requesting cycle sequences.

6.2.2. Quantifying the Gross Interests

Based on the previously introduced requesting model, a quantification of Gross Interest in terms of the expected number of active clients (those who perform actively requests to the service with the maximum possible requesting rate) is proposed.

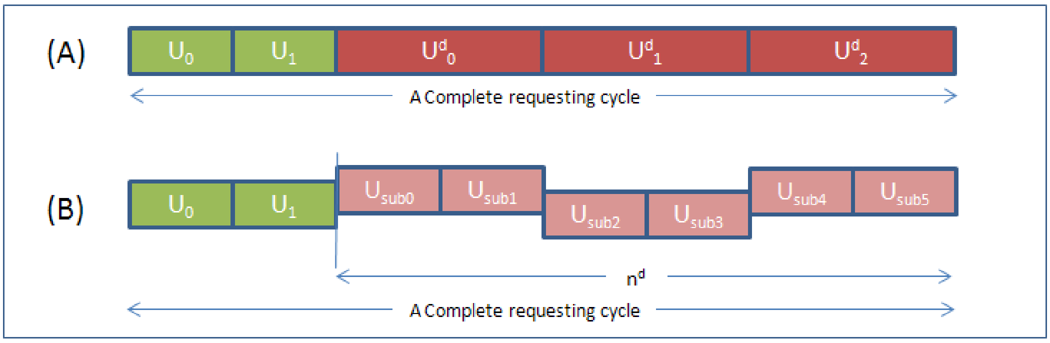


Figure 6.2.: One complete requesting cycle

For simplicity of derivation, we assume that the requesting model specifications (U, U^d, N, M and $RequestRate$) are fixed for all the requesting cycles.

As shown in Figure6.1-B, let U be the time length of a requesting unit. Let n be the number of requesting units. Let U^d be the time length of a pausing unit. Let m be the number of the pausing units.

Case 1: In this case we assume that $U = U^d$. The Sum of all calling and pausing unites will be $m + n$. The probability for a client to be in an active period at a certain time $Prob(active)$ will be:

$$Prob(active) = \frac{n}{n+m} \tag{6.1}$$

Considering the introduced requesting rate in our requesting model, since $RequestRate \in \{outcome(distribution(Rates))\}$, the probability for a client to achieve the maximum requesting rate at a certain time $Prob(active_{MaxRqst})$ will be:

$$Prob(active_{MaxRqst}) = \frac{n}{n+m} \times P(MaxRqst) \tag{6.2}$$

where $P(MaxRqst)$ is the probability for a client to have a $RequestRate =$ maximum request rate and $outcome(\ distribution(Rates)\ )$ is a function that produces the possible requesting rates. Therefore, the expected number of active clients who perform the maximum allowed requesting rate at a certain time Ex will be:

$$Ex = NetworkSize \times \frac{n}{n+m} \times P(MaxRqst) \tag{6.3}$$

Case 2: Here, $U \neq U^d$ (see Figure 6.2-A and B). U^d units are substituted by U units. Therefore, let us define R, which is the ratio of U^d to U where $R = \frac{U^d}{U}$ and so we substitute U^d by U. The new number of the new substituted U units is n^d

and n^d replaces m where $n^d = R \times m$. The summation of all active and pause periods will be $n + R \times m$. For this Equation 6.2 should be modified to:

$$Prob(active_{MaxRqst}) = \frac{n}{n + R \times m} \times P(MaxRqst) \tag{6.4}$$

and so Ex will be:

$$Ex = NetworkSize \times \frac{n}{n + R \times m} \times P(MaxRqst) \tag{6.5}$$

Since n and m are based on some distribution as mentioned before, they should be used as their expected values. Let Ex_n be the expected value of n and Ex_m be the expected value of m. So Ex will be:

$$Ex = NetworkSize \times \frac{Ex_n}{Ex_n + R \times Ex_m} \times P(MaxRqst) \tag{6.6}$$

Based on the previously mentioned derivations the Gross Interest can be defined.

Definition 3 *: The Gross Interest*
The expected number of active clients connected to a specified service and performing the maximum allowed requesting rate at a certain time.

Correlation to Service Popularity: Converging to the concept of service popularity, the expected Gross Interest can represent a common reference point. If the service providers know (based on experience, questionnaires, ...etc.) how the clients may access their services, based on the proposed requesting model, they can present an approximate for the Gross Interest regarding an approximate for the Gross Interest regarding some services. Moreover , the Gross Interest for many services and service categories may be known in advance. In case of presence of competition among many providers who operate services with the same functionality in the same network, a provider can estimate how much interest is gained at his offered service. Not only from a perspective of a provider but also upon the level of the network, these functionally equivalent services can be ranked reflecting their aggregated popularity.

6.2.3. Criteria of The Generalized Requesting Model

The properties of the proposed generalized requesting model can be summarized as follows:

- Easy and clear abstract numeric based attributes to describe the requesting behavior of the clients. The quantification process of the requesting model can be generalized and different requesting specifications can be aggregated in the same computation style.

- Heterogeneity of specifications: Each requesting behaviors of the interested clients can be modeled individually. The resultant (maximum) Gross Interest is computable. Then, at the provider side, the number of requests that are really gained at any provider can reflect the relative popularity of the offered service. Moreover, the specifications are time varying which adds a dimension for the variant vitality (importance degree) of a service from a client perspective over the operation time of the network.

- Subsection 6.2.2 shows that, as the number of the network participants increases, higher number of clients are active. In some environments like web services in the Internet, the network size is huge. Therefore, it is more suitable, in these cases, to tune down the weight of the network size in the computation of Gross Interest.

6.3. Gross Interest Scenarios: Specifications and Quantifications

If service providers care about the popularity of their offered services, they have to think firstly about estimating their offers' popularity. In this section, we give two extremely different sets of specifications for the proposed generalized requesting model. Based on the proposed computation method of the Gross Interest in Subsection 6.2.2, the expected number of active clients is computed as an indication for the service popularity. We care about the simplicity of the specifications in both of the proposed scenarios. Therefore, as derived in Subsection 6.2.2, the specifications are assumed to be fixed for all the service clients during the network operation time. Both scenarios represent a pilot applications for the previously presented concepts of the service popularity, Gross Interest, and the generalized requesting model.
In order to quantify the Gross Interest in both scenarios, both the network size and the requesting rate should be specified. The network size may indicate the real number of the network participants or the expected size of the network participants who are targeted by a specified service. The network size varies from small numbers like in mobile ad hoc networks to very large numbers like in the Internet. As previously introduced, the weight of the network size needs to be tuned in some cases. On the other hand, a suitable statistical distribution for the request rate should be specified. In our investigations, in order to have an estimation, we set the network size to be 25 participants and the requesting rate to be uniformly distributed between 0 and 3 requests per minute.

6.3.1. Scenario 1: Rich Gross Interest

In this Gross Interest scenario, a high number of frequent requests regarding a specific service is modeled as follows:

- U is 10 minutes.
- N is uniformly distributed between 1 and 5 units.
- U_d is 3 minutes.
- M is uniformly distributed between 0 and 4 units

Therefore, the expected number of E_x is about 5.2 clients a time, which is a high number compared to the network size.

6.3.2. Scenario 2: Poor Gross Interest

In contrast to Scenario 1, this scenario models a low number of infrequent requests regarding a specific service as follows:

- U is 2 minutes.
- N is uniformly distributed between 1 and 3 units.
- U_d is 10 minutes.
- M is uniformly distributed between 0 and 4 units

Therefore, the expected number of E_x is about 1.0 clients a time, which is a low number compared to the network size.

Comparing Scenarios: The specifications lead to long active periods in the first scenario versus short active periods in the second scenario and short pause periods in the first scenario versus long pause periods in the second scenario. The two Gross Interest scenarios produce two extreme (high and low) number of clients expected to be active a time. The resultant expected number of clients for both scenarios indicates a higher service popularity in the first scenario. If a normalization between the two expected values can be realized, then the first scenario describes a very popular service while the second scenario describes a service with less interest by the client set.

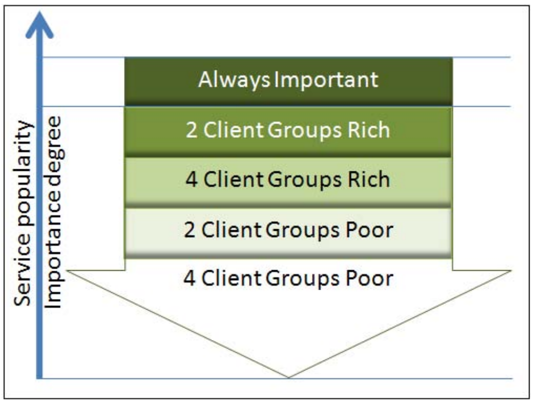


Figure 6.3.: Service popularity ranges: combination of requesting scenarios and different requesting rates

6.3.3. The Role of the Requesting Rate: Gross Interest Ranges

One of the most important features of the proposed generalized requesting model it the requesting rate. Specifying the requesting rate can provide a service popularity categorization. By varying the requesting rate, different service popularities can be modeled. The clients of a specified service can be categorized into groups (*client groups*) based on the different specified requesting rate for the same Gross Interest scenario. Figure 6.3 illustrates this categorization. If the requesting rate varies between 0 and 1 requests per minute, this means that, at any time of the network operational time, there are two groups of more or less interested clients. The first group of client has clients with 0 requests per minute and the second group has clients with 1 requests per minute. In this category and based on the previous derivations in Subsection 6.2.2, E_x will be about 10.4 clients in the rich scenario and about 2.08 clients in the poor scenario. If the requesting rate varies between 0 and 3 requests per minute, there are four groups of clients. The first group of client has clients with 0 requests per a minute, the second group has clients with 1 requests per minute, the third group has clients with 2 requests per minute, and the fourth group has clients with 3 requests per minute. In this category, the number of E_x will be about 5.2 clients in the rich scenario and about 1.04 clients in the poor scenario.

So, as depicted in Figure 6.3 and based on any proposed relative normalization for applying these client groups and in which scenario, we obtain here five different service popularities. On top of the proposed popularity ranking arrow, the "always important" services. This category contains those services which are vital for the network participants during the entire operation time of the network. Therefore,

estimating the popularity for this category is not meaningful. Based on the computed Gross Interest, the order of the popularity of the rest of four categories are the "2 Client Groups Rich", "4 Client Groups Rich", "2 Client Groups Poor", then "4 Client Groups Poor".
The previous categorization can be done based on any other sets of specifications for the generalized requesting model. This categorization represents a practical try for ranking the popularity of services. By using such a categorization, a different requesting behavior has been modeled regarding a specified service or a set of services with same functionalities.

6.4. Enhancing The Service Content

Service Content: What is meant by "service content" is the collection of services and replicas needed to provide the required functionalities in the interest scope of the network participants of a specified network at a specified time. Based on the type of services and their available resources, all the network participants can be service providers. Each of these proposed service providers can provide arbitrary services. The important question about the network performance is answered in terms of how the network participants access these services (in terms of some metrics like amount of transported data). The service popularity can play a great role here. The main concerns about what is important to be offered can be explained in terms of modeling the client requesting behavior and popularity. Involving the service popularity in the process of enhancing the service content is important.
In this section, the advantages which can be gained by applying any scheme (like the Gross Interest scheme) to grade and rank the service popularity are discussed. These advantages can be explained from different perspectives: (a) From a client perspective, based on semantics, the popular services' popularity can reflect a positive or negative effect on the service trust. Moreover, popular services are more likely to provide consistent results regarding the client requests, since they have been mostly and correctly called from a higher number of the society members. (b) From a service provider's perspective, by computing the Gross Interest of the proposed generalized requesting model, a service provider obtains an estimate of how often his service is called relative to the total Gross Interest. As discussed before, it can be used as a measurement to indicate whether it is necessary to deploy a new version of a specified offered service. (c) From a network perspective, utilizing the network resources for the offered services can be saved. For example a service with longer process enforces the clients to keep their connected sessions to this service longer and then their resources are occupied longer. If the service providers are switching down their uninteresting services, the service repositories (the network parties who are responsible for collecting the service offers) will con-

tain a smaller number of service offers. Finding the required services offers in this case will require a smaller searching effort and shorter time. If such an election (based on popularity) is dynamically done, the network will be enabled to preserve the required service contents and only those.

Requirements' Index: The Requirements' Index has been introduced as a service attribute in Chapters 4 and 5. The objective behind this attribute is to combine the service popularity to the volume of requirements (of resources) required to enable a client to host a replica of a specified service. Moreover, not only the functional requirement or resources are assumed to be combined to the service popularity by the Requirements' Index but also some non-functional attributes can be combined like fees for the forwarded replica or some operational constraints like the number of connected sessions (see Chapter 11). During the rest of this work, the Requirements' Index is generated under some distribution (normally) and labels the allocated replicas to give different grades and determine the overall replication cost. Regarding the core functionalities of the replication mechanism as discussed in Chapter 5, the Requirements' Index represents the cost of "Pass a replica" functionality (see Subsection 4.2.1).

6.5. Summary

In this chapter, the importance of the service popularity as a non-functional attribute of services and its related issues have been addressed and discussed. The Gross Interest has been introduced as a proposed measurement for the service popularity. As previously mentioned, since the Gross Interest is based on a set of client's requesting behaviors regarding a specified service, a generalized requesting model that can capture a client requesting behavior has been introduced. Two sets of specifications (scenarios) for the proposed generalized requesting model have been described. A quantification method of the Gross Interest for these two groups of specifications showed how these scenarios are quite different from a service popularity perspective. More investigations have been given to highlight the role of the requesting rate of the proposed generalized requesting behavior. Finally, advantages of having a service popularity consideration in a service-based application have been presented and discussed in terms of "service content" and the Requirements' Index.

Further motivation questions:

This chapter answered the question how to configure the requesting behavior and the request generation process regarding a service and introduced a popularity-based method that can specify the client requesting behavior regarding services in the rest of this thesis. At this stage of our work, the motivation question is:

- What are the effects of the replication and hibernation behaviors on SDP performance?

This question of the proper set of specifications for SDP replication and hibernation behaviors is addressed and investigated during the next chapter.

The suggested Gross Interest scenarios appear with different names in the next chapters. The terms of “rich vs. poor” and “scenario 1 vs. scenario 2” are used interchangeably to indicate the suggested Gross Interest scenarios.

CHAPTER 7

Effects of Different Replication/Hibernation behaviors on SDP

"I constantly sought knowledge and truth, and it became my belief that for gaining access to the effulgence and closeness to God, there is no better way than that of searching for truth and knowledge"

(Alhazen)

Since the two main SDP mechanisms of replication and hibernation are service popularity based and since these two mechanisms are triggered based on the requesting behavior and the related replication and hibernation thresholds, the motive for this chapter is to determine what the effects of the different configurations of the thresholds on the service distribution process are. The Gross Interest scenarios, introduced in Chapter 6, are applied here. The correctness of the replica allocation (placement) process is modeled in different ways depending on the resultant number of replicas inside a given partition. Moreover, an extended analysis that aims to figure out the number of produced replicas inside a specific network partition is presented. For that, we introduce the concept of the "Typical Partition". The simulation results provide a detailed view on the performance of SDP and how it behaves versus different and challenging constraints and specifications. The work of this chapter has been partially published in [HKR09c, HKR08b].

The structure of this chapter is as follows: Section 7.1 presents the allocation correctness ratios for the service distributions. In Section 7.2, the proposed replica allocation correctness ratios of Section 7.1 are evaluated in details using simulations. Finally, the work of this chapter is summarized, the contributions are highlighted, and the next research motivations are presented in Section 7.3.

7.1. Correctness of Replica Allocation

As previously introduced in Chapter 5, the number of mobile hosts with active replicas is the main parameter of determining the correctness of the replica allocation process. The correctness of a replica allocation should indicate the relation between the number of active replicas to the size (number of mobile hosts) of a network partition at a specific time. Thus, we have defined in Section 5.2 defined two correctness ratios: The *Linear Correctness Ratio*(LCR_t at a specified time t) and the *Rational Correctness Ratio*(RCR_t at a specified time t).

The evaluation will show that this straightforward linear computation method has deficiencies. In particular, it is not sensitive to the partition size, since the computing of the average value is giving the same weight to different partition sizes, and it is not sensitive to the number of replicas in cases of large partition sizes.These deficiencies motivated us to find more expressive measures for the correctness of the replica allocation process. The *Weighted Linear Correctness Ratio* $WLCR$ is the weighted sum of the linear allocation correctness ratios in the different partitions. This computation method is sensitive for the size of the network partition, where ($WLCR_t$) at a given time t is:

$$WLCR_t = \sum_{i=0}^{n} (\frac{Pz_i}{NetworkSize} \times LCR_t(P_i)) \tag{7.1}$$

As previously discussed in Chapter 5 RCR is more sensitive for higher numbers of replicas inside a partition. For example, with a partition size of 50 nodes, the allocation correctness ratio of 20 replicas will be 0.09. On the same style of $WLCR$, the *Weighted Rational Correctness Ratio* ($WRCR$), which is the weighted sum of the rational allocation correctness ratios in the different partitions, is defined. It is sensitive for both the partition size and the higher numbers of replicas inside the same network partition. On a rational basis to the number of active replicas $WRCR$ can be computed as follows:

$$WRCR_t = \sum_{i=0}^{n} \left(\frac{Pz_i}{NetworkSize} \times RCR_t(P_i) \right) \tag{7.2}$$

During the next sections of this chapter, the four proposed correctness ratios will be used to estimate the optimality of the generated service distributions of SDP. The advantages and disadvantages of these computation methods will b analyzed versus different replication and hibernation specifications through the proposed evaluation experiments.

7.2. Evaluation

The evaluation depends on an extensive simulation based on an extensive simulation tool (for more details, see Appendix A). The proposed results are divided into two main categories: first, the general performance of SDP is evaluated against different replication/hibernation behaviors, then effects on the different correctness ratios are investigated.

7.2.1. Performance Parameters and Settings

The network maintains a service under the same constraints defined in Section 5.1. The simulation run time for each experiment was set to be 2 hours. The network is placed in a $600m \times 600m$ square area, varying its size from 10 to 140 mobile hosts. Each mobile node can cover a transmission range with a fixed radius of 75 meters. The Random Waypoint [LNR04] mobility model is applied to the mobile nodes. The maximum allowed speed for the mobile nodes is $6m/s$ with a maximum residence (pause) time equal to 15 minutes. Each plotted point in the presented curves comes from the average of 20 runs. The two proposed Gross Interest scenarios in Chapter 6 (rich: Scenario 1 and poor: Scenario 2) have been used as the requesting behavior for the clients here. The number of client groups is either 2 groups (maximum request rate 1 request / minute) or 4 groups (maximum request rate 3 requests / minute). The replication threshold for a client is achieving a number of requests equal to the maximum request rate (Max-request-rate:1). The hibernation threshold for a provider is gaining less than one request in a given length of minutes. This given interval will be either 1, 3, or 5 minutes. The performance parameters in this evaluation are the service availability, success ratio, prevalence, residence time, and replication degree as previously introduced and defined in Chapter 4 and 5.

The concepts of the service prevalence, replication degree, and the replica allocation correctness ratio are closely related and could be interpreted as different terms for indicating the number of running replicas inside the network. Each of these parameters has its own perspective in highlighting the number of the concurrently running replicas: while the service prevalence measures that number on the scale of the whole network, the replication degree summarizes the status of the service distribution in the network on the level of the network partition. The replica allocation correctness ratios represent a judge how close each of the network partition is near to the optimum assumed distribution in a given time.

7.2.2. Primary Evaluation

As shown in Figure 7.1, in Scenario 1, the service availability increases as the network size increases and decreases as the number of the client groups increases for the three hibernation thresholds. This is due to the smaller set of the participants ready to host a replica at a time because, as one of the settings, the replication threshold is set to be the value of the maximum request rate in terms of number of requests per a minute. In case of 4 groups, just one quarter of the participants can achieve that requesting rate. On the other hand in the case of just two groups, half of the participants can achieve the maximum requesting rate . The same behavior can be noticed in Scenario 2, but the growing of both of availability and success ratio are slower and later. This is due to the poor Gross Interest in this scenario.

Although keeping longer time intervals for the hibernation threshold is better to keep higher service availability and success ratio, the effect of shortening the hibernation threshold has a relative small effect on both availability and success ratio for both scenarios. Table 7.1 clarifies these differences.

Figure 7.5 shows the prevalence ratio of the network participants. For Scenario 1, the higher the number of client groups the lower the prevalence ratio. The effect of longer hibernation thresholds increases the prevalence ratio. The notable result here is the curve of the (1 : 1) hibernation threshold, it seems to be usually steady under 0.20. On the other hand, in Scenario 2, the same behavior can be noticed for the curves of the different hibernation thresholds for both client groups. However, the effect of the poor Gross Interest makes the prevalence ratio very low.

The service residence time decreases as the hibernation threshold test interval increases and as the network size increases for all client groups and both scenarios. As the number of the client group increases, the residence time increases. This is

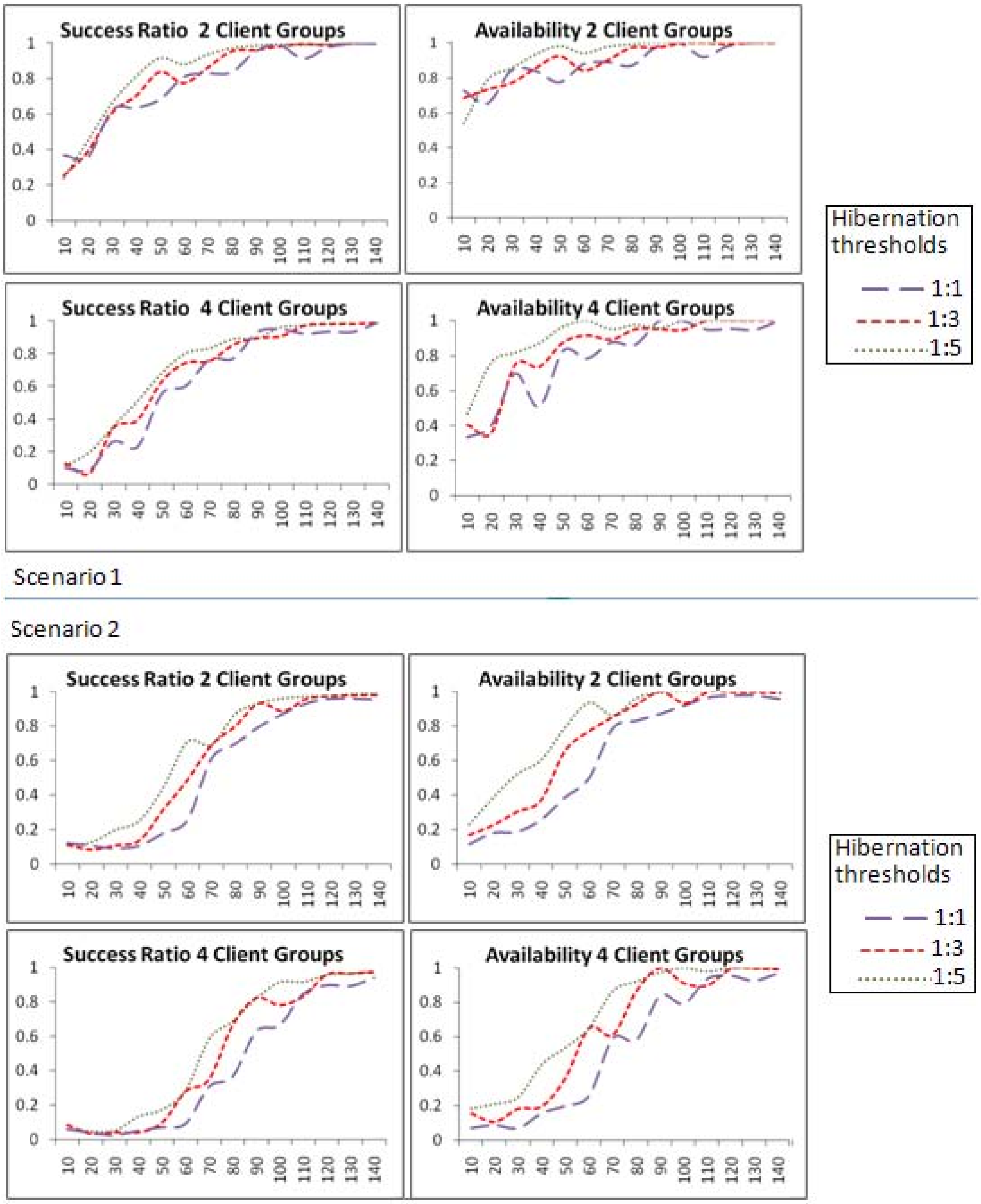


Figure 7.1.: Service availability and success ratio against network size for Scenario 1 and 2

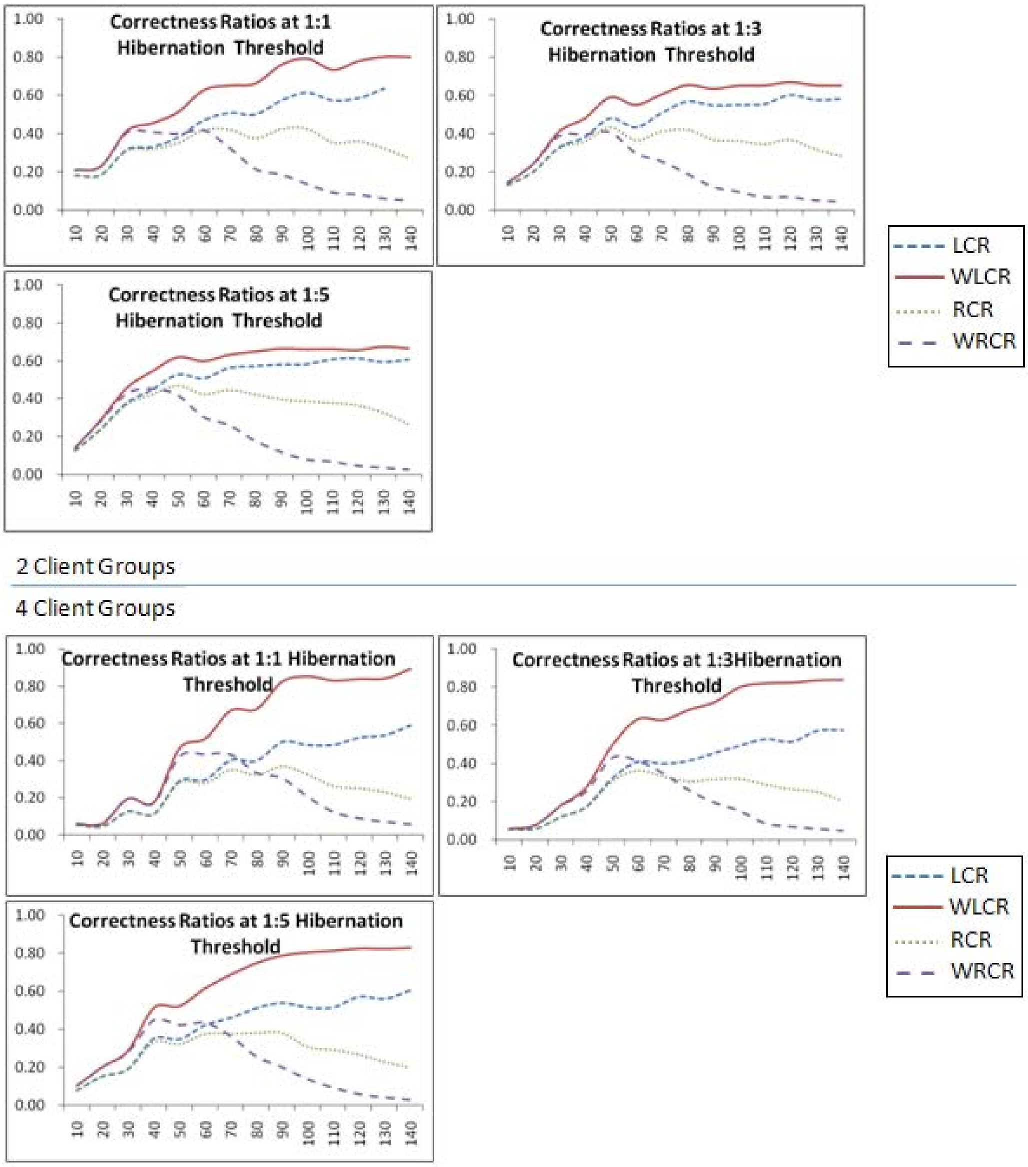


Figure 7.2.: Different allocation correctness computations against network size for Scenario 1

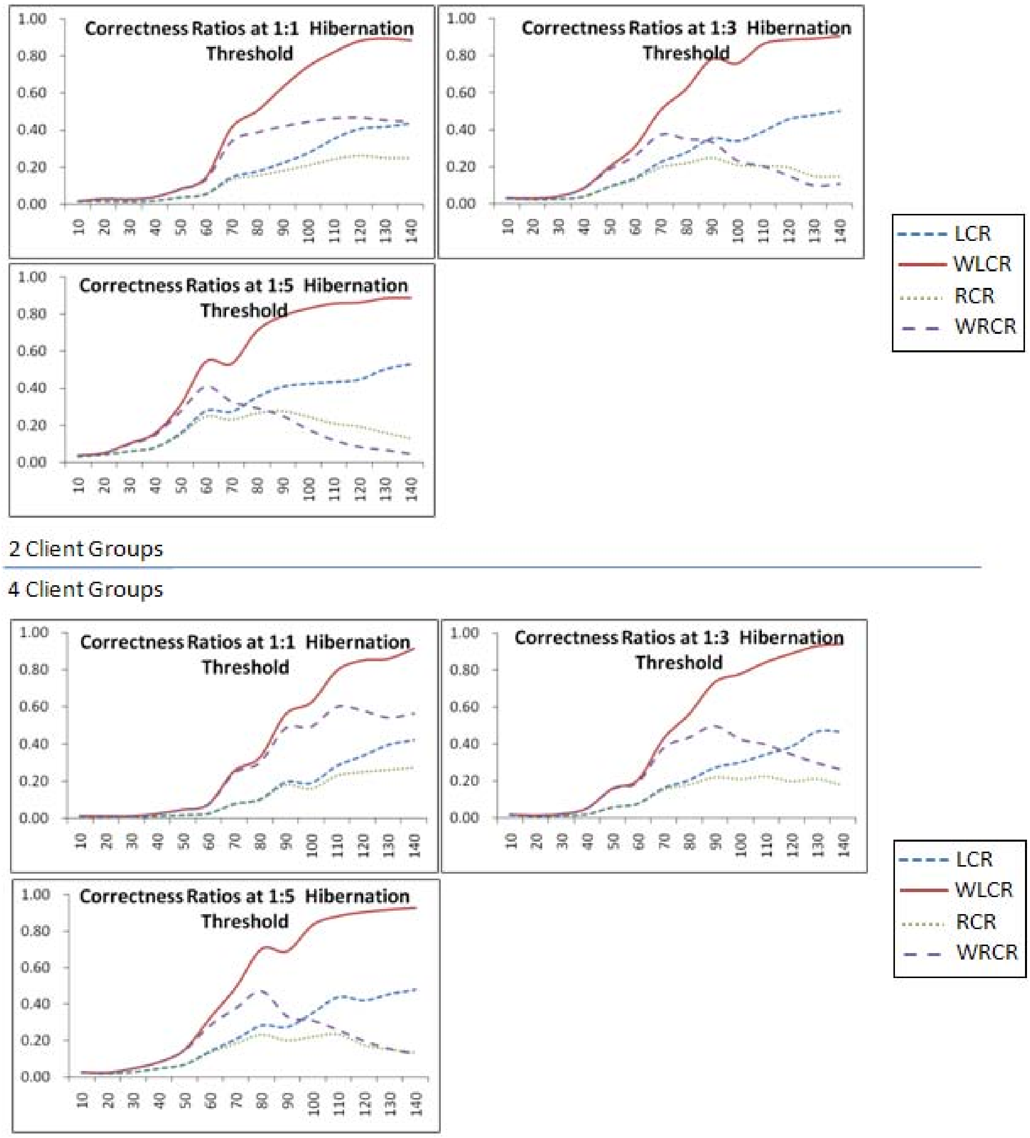


Figure 7.3.: Different allocation correctness computations against network size for Scenario 2

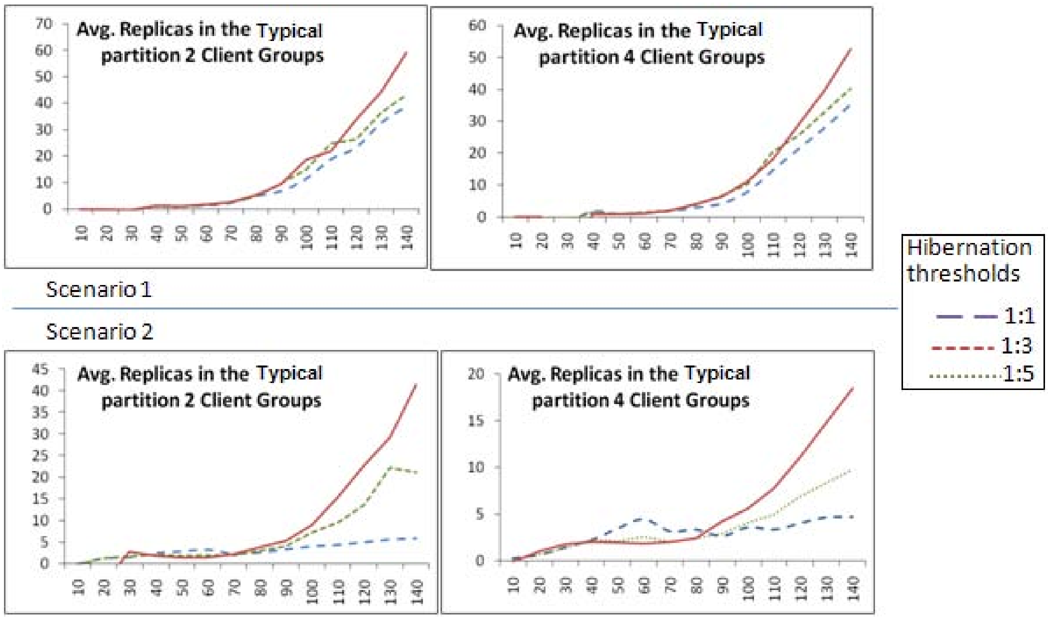


Figure 7.4.: Number of replicas in the Typical Partition against network size for Scenario 1 and 2

very logical because as the number of client groups increases, the number of clients interested to have a replica will decrease since the replication threshold will be set to the higher request rate which will be assigned for a fewer number of the clients. As a clear note, this effect is reduced as the network becomes more dense. This is logical, too, because the number of the clients interested to host a replica starts increasing in denser networks.

Figure 7.6 shows the resultant replication degree of the protocol in both proposed scenarios with respect to the number of client groups. The effect of the Gross Interest type is clear. The replication degree varies between about 0.3 in the rich gross scenario with 2 client groups to less than 0.05 in the poor Gross Interest scenario for 4 client groups, as in figure7.6. The replication degree increases as the Gross Interest increases and number of client groups decreases. This result confirms our concept that the more interesting services should prevail and stay more in the network. The average values and the related standard deviation of Figures 7.5, 7.6 are presented in Table 7.2.

In Figure 7.7 the resultant linear replica allocation correctness is examined against different hibernation behaviors. As seen, the linear allocation correctness increases as the network size increases. In Scenario 1, the different resultant correctness ratios reach about 0.6 as maximum ratios at starting from about 50 nodes network

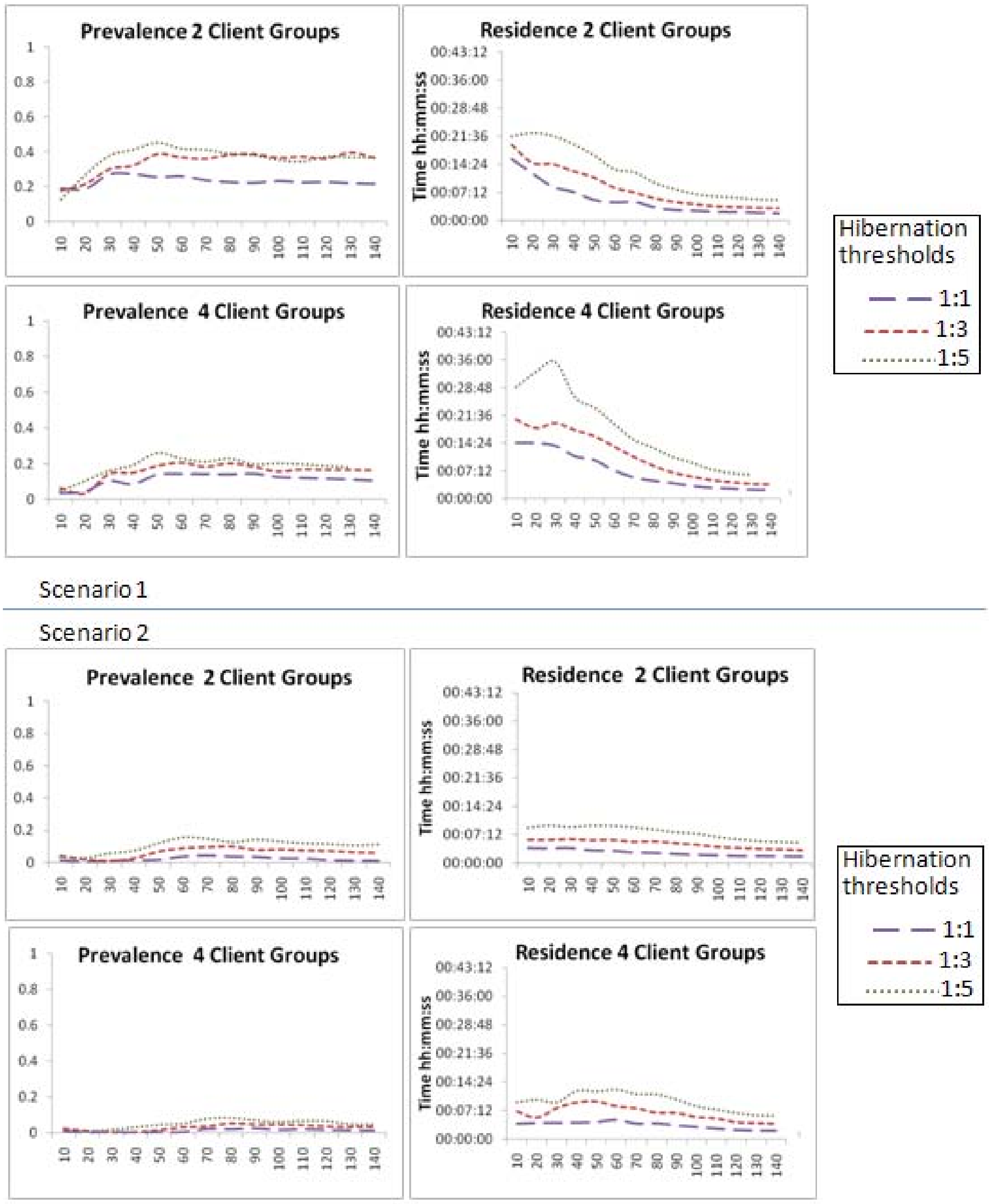


Figure 7.5.: Service prevalence and residence time against network size for Scenario 1 and 2

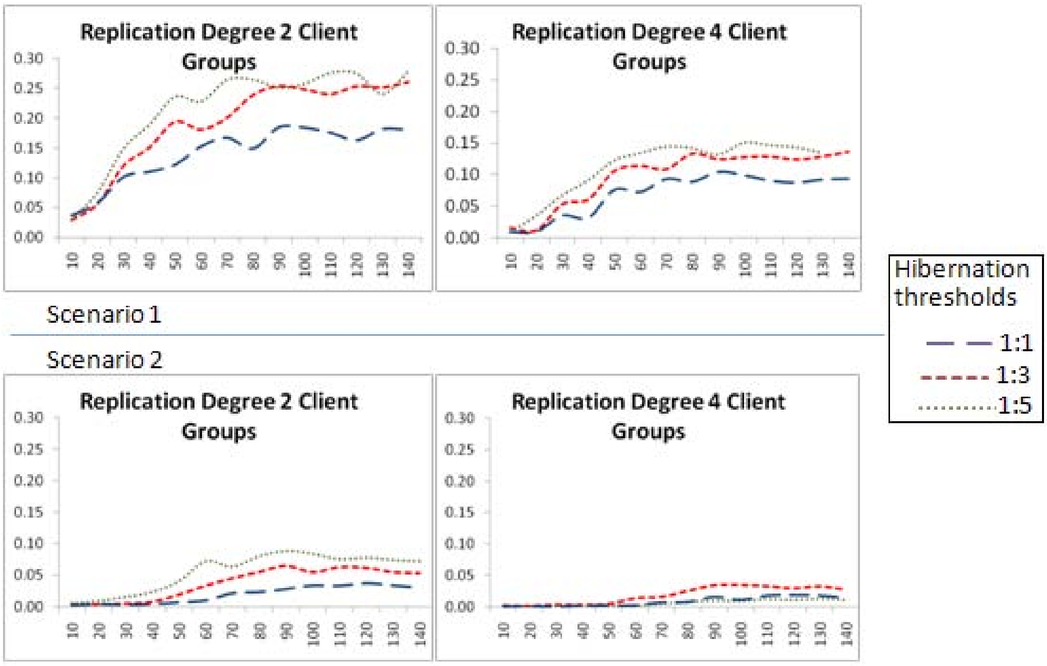


Figure 7.6.: Replication degree of both numbers of client groups for Scenario 1 and 2

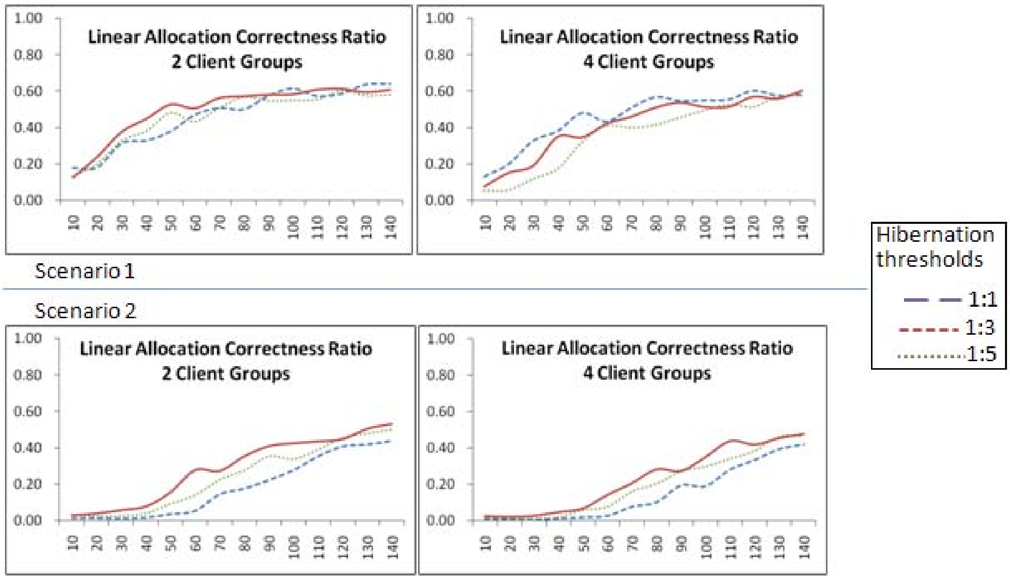


Figure 7.7.: Linear allocation correctness against network size for both Scenario 1 and 2

			Success Ratio		Availability	
		Hib.Thr.	Avg.	Avg.StD.	Avg.	Avg.StD.
Scenario1	2Groups	1:1	0.79	0.21	0.88	0.19
		1:3	0.81	0.15	0.90	0.16
		1:5	0.85	0.11	0.92	0.11
	4Groups	1:1	0.64	0.17	0.80	0.24
		1:3	0.69	0.15	0.84	0.21
		1:5	0.70	0.12	0.91	0.15
Scenario2	2Groups	1:1	0.55	0.11	0.64	0.13
		1:3	0.60	0.10	0.73	0.13
		1:5	0.66	0.11	0.81	0.12
	4Groups	1:1	0.42	0.12	0.53	0.17
		1:3	0.49	0.11	0.64	0.15
		1:5	0.54	0.10	0.71	0.16

Table 7.1.: Average (Avg.) success ratios and availability with the related standard deviation (Avg.StD.) against different hibernation thresholds

			Residence		Prevalence		Replication Degree	
			Avg.	Avg.StD	Avg.	Avg.StD	Avg.	Avg.StD
Scenario1	2 Groups	1:1	00:05:21	00:01:57	0.233	0.086	0.140	0.049
		1:3	00:08:14	00:02:19	0.341	0.088	0.191	0.057
		1:5	00:12:23	00:02:58	0.361	0.082	0.215	0.050
	4Groups	1:1	00:07:05	00:03:41	0.111	0.055	0.070	0.029
		1:3	00:10:59	00:05:33	0.156	0.063	0.098	0.035
		1:5	00:18:11	00:07:11	0.186	0.059	0.112	0.035
Scenario2	2 Groups	1:1	00:02:35	00:00:34	0.023	0.018	0.019	0.007
		1:3	00:04:56	00:00:44	0.063	0.031	0.038	0.013
		1:5	00:07:39	00:00:56	0.105	0.044	0.056	0.019
	4Groups	1:1	00:03:31	00:01:26	0.014	0.012	0.009	0.005
		1:3	00:06:29	00:01:44	0.030	0.020	0.019	0.009
		1:5	00:09:25	00:02:08	0.049	0.027	0.007	0.004

Table 7.2.: Average (Avg.) residence time, prevalence, and replication degree with the related standard deviation (Avg.StD.) against different hibernation threshold

size for 2 and 4 client groups. On the other hand, the correctness ratio is slower in growing and requires higher dense situation of the higher network sizes to reach about 0.5 for 2 and 4 client groups.

These results have shown the deficiencies of the linear computation method for the allocation correctness. Also, in shorter hibernation thresholds, like (1 : 1), the uninteresting replicas should be terminated quicker than with other longer threshold. This quick hibernation should be reflected as higher allocation correctness which could not be seen or interpreted by these results to summarize deficiencies are:

- Not sensitive to the partition size, since the computing of the average value gives the different partition sizes the same weight.
- Not sensitive to the number of replicas in cases of large partition sizes.

7.2.3. Further Evaluation

In this subsection, we evaluate the different allocation correctness methods (LCR, WLCR, RCR, WRCR) as a measure for the optimality of the replica placement process of the service distribution protocol.

As shown in Figure 7.2 in Scenario 1, the higher the number of client groups, the higher the ratios of LCR and WLCR. In a large number of partition sizes and higher allowed number to host replicas (replication threshold), both LCR and WLCR are directly proportional to the size of the network. At higher numbers of client groups the difference between LCR and WLCR is noticeable. On the other hand, both RCR and WRCR start increasing during a certain interval of network sizes then the curves reflect their behavior and start a slow (for RCR) or quicker collapsing. The wideness of this interval of the network sizes depends not only on the length of the hibernation evaluation time interval but also on the number of client groups. As the hibernation threshold increases, the wideness of the stable RCR and WRCR observed interval decreases, and as the allowed number of participants allowed to host replica decreases this interval decreases as well.

In case of Scenario 2, with the poor Gross Interest and as shown in Figure 7.3, the effect of the low number of produced active replicas inside the different partitions is clear. The growing of all the curves is very slow until a certain network size and then takes a similar behavior to those of Figure 7.2 except the WRCR at hibernation threshold (1 : 1) for both client groups. As the network size increases, WRCR increases (in contrast to all the other WRCR curves in the other cases). This contradiction can be explained as follows: The shorter hibernation threshold enforces the concurrent active replicas to be hibernated quicker. Since the available number of the active replicas is really small (regarding the poor Gross Interest in Scenario 2), the active number of replicas will be smaller too. The other main parameter affected by that behavior is the number of the client groups, as the replication threshold, as the large number of participant allowed to host an active replica decreases, the average WRCR ratio increases (from about 0.4 starting from 80 nodes in 2 client groups to about 0.6 in 4 client groups). Table 7.4 shows the average vales of the four allocation correctness ratios and the related resultant average standard deviation.

As general notes, first, the differences between the three proposed computing ratios are very clear: while WLCR is telling that everything is very nice (in Scenario 1, values of the allocation correctness reach about more than 0.9 in some cases like at hibernation threshold (1 : 1) in 4 client groups starting from about 80 nodes network size), both RCR and WRCR are saying something very different (in Scenario 1, values of the allocation correctness reaches about less than 0.35 and decrease in the same case of at hibernation threshold (1 : 1) in 4 client groups starting from about 80 nodes network size). This difference is due to the sensitivity of both RCR and WRCR to the number of the replicas inside each partition compared to both LCR and WLCR.

Second, in Scenario 2, as we can see in Figure 7.3, no allocation correctness ratios can be achieved until higher network sizes are available which ensure smaller number of formed network partitions.

In the following subsection, the issues of identifying and sampling the partitioning behavior of the network and deducing the related produced number of replicas inside these sample partitions are shown.

7.2.4. The Typical Partition

As mentioned in Chapter 6, the service popularity may rise as the network size increase. In MANETs, since the network is ever-partitioning, the influence of the network size becomes expressed by the network partition size. The relation between the network size and the partition size is determined by the mobility behavior of the network participants. In this chapter, estimating how the network participants are concentrated in the different network partitions reflects the needs to estimate the number of active replicas inside the most weighted network partition (with respect to the partition size).
Let ($Typ.Partition_{Net}$) be the Typical Partition of a given network size $size(Net)$ consisting of n partitions where the i_{th} partition size is $size(P_i)$ at a given time t , then

$$Typ.Partition_{Net} = \sum_{i=1}^{n}(size(P_i) \times \frac{size(P_i)}{size(Net)}) \tag{7.3}$$

By observing the size of the Typical Partition, as a sample for the partitions of this network, and using the measured WRCR ratio, the number of the currently active replicas inside the Typical Partition can be computed. Figure 7.8 shows the number of network partitions and the size of the Typical Partition against the network size. The number of partitions observed in this evaluation was varying between about 8 partitions for the network size of 10 nodes. It becomes about 15 nodes in the networks-sizes interval of $[40, 70]$ nodes, then it decreases to be about less than 3 network partitions at network size of 140 nodes. As the network size increases, the size of the Typical Partition increases too and indicates that at higher network density, most of the network participants belong to the same partition most of the network operation time. For example, at network size of 120 nodes, the observed important size is about 98 nodes.

Figure 7.4 shows the computed number of replicas inside the Typical Partition for both scenarios versus different settings and against the network size. The most important results are, if we consider two intervals for network sizes: the first interval is the “low-moderate” interval which includes network sizes less than or equal to 90 nodes. The other interval contains the “higher” network sizes. Table 7.3 shows the difference between the average number of replicas per each of the low-moderate interval “Low.Int.” and the higher interval “High.Int” network sizes. Generally, the number of the generated replicas inside the Typical Partition is usually less than 3 active replicas in the low-moderate interval against all of the other settings. This result is considered to be very important. In general, the number of the concurrently running active replicas is small. The proposed protocol can for lower and moderate network sizes (till 90 nodes) produce a very limited number of concur-

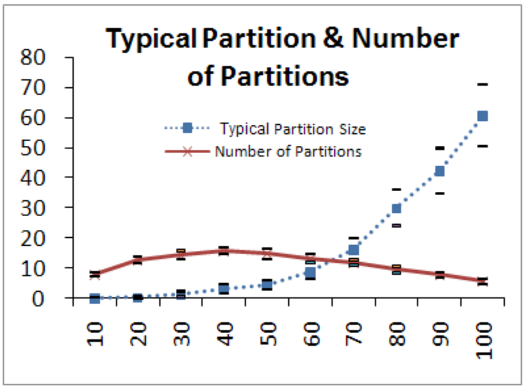


Figure 7.8.: Size of the Typical Partition and the number of partitions against the network size

		Hib.Thr.	Nr.Replicas Low.Int.	Nr.Replicas High.Int.
Scenario1	2Groups	1:1	1.87	24.91
		1:3	2.27	29.22
		1:5	2.29	35.58
	4Groups	1:1	1.25	21.43
		1:3	1.66	25.95
		1:5	0.23	30.17
Scenario2	2Groups	1:1	2.18	4.97
		1:3	1.94	14.66
		1:5	1.51	23.45
	4Groups	1:1	2.40	4.06
		1:3	2.03	6.80
		1:5	1.89	11.56

Table 7.3.: Average number of the computed concurrent active replicas inside the Typical Partition

			LCR		WLCR		RCR		WRCR	
		Hib.Thr.	Avg.	Avg.StD.	Avg.	Avg.StD.	Avg.	Avg.StD.	Avg.	Avg.StD.
Scenario1	2Groups	1:1	0.46	0.14	0.60	0.15	0.34	0.13	0.23	0.11
		1:3	0.46	0.12	0.54	0.11	0.33	0.11	0.20	0.08
		1:5	0.50	0.10	0.57	0.08	0.36	0.10	0.20	0.07
	4Groups	1:1	0.35	0.12	0.56	0.15	0.23	0.10	0.21	0.11
		1:3	0.36	0.12	0.56	0.13	0.24	0.11	0.19	0.10
		1:5	0.42	0.10	0.61	0.09	0.28	0.09	0.22	0.09
Scenario2	2Groups	1:1	0.18	0.05	0.44	0.08	0.13	0.04	0.27	0.06
		1:3	0.24	0.07	0.49	0.08	0.14	0.05	0.18	0.07
		1:5	0.29	0.07	0.54	0.08	0.16	0.06	0.17	0.07
	4Groups	1:1	0.15	0.06	0.38	0.10	0.11	0.04	0.28	0.09
		1:3	0.20	0.07	0.47	0.10	0.13	0.05	0.25	0.08
		1:5	0.23	0.07	0.50	0.09	0.13	0.05	0.20	0.08

Table 7.4.: Different (Avg.) allocation correctness ratios with related standard deviation (Avg.StD.)

rently active replicas. On the other hand, and for the high network densities in the higher intervals of sizes starting form 100 nodes, the number of produced replicas is directly proportional to the network size. It grows faster as the hibernation threshold increases. As the number of the client groups increases, the number of the generated replicas decreases. In Scenario 2, the number of the concurrent active replicas in general is less than in Scenario 1.

7.3. Summary

In this chapter, we showed that the application and effects of the two extremely different proposed Gross Interest scenarios in Chapter 6. The effects of different replication thresholds and hibernation thresholds (behaviors) on the general performance and the replica allocation process are investigated and discussed. Based on different computation schemes for the correctness of the replica allocation process, the advantages and disadvantages of these proposed correctness ratios were discussed. Finally, an extended analysis for indicating the number of concurrently active replicas inside a sample network partition (Typical Partition) was presented and shown how the concurrently running number of active replicas is really limited for most of the configurations, especially for low to moderate network sizes.

Further motivation questions:

As presented in the results, as the hibernation test intervals becomes shorter, better allocation correctness can be achieved. The key motivation questions at this stage of our work are:

- What is the optimum minimum hibernation test interval?
- How can these "optimum" time intervals be determined?

These motivation questions are going to be answered in the following chapter in which we propose mechanisms to enable SDP to manage different time lengths for the hibernation test interval dynamically.

Since WRCR is more restricted and compatible for our assumptions in Section 4.7 where only one service or replica inside a network partition can serve all of its participants, further in this research, the defined WRCR ratio in Section 7.1 is used as the indication for the replica allocation correctness unless something else is stated.

CHAPTER 8

Leader Election Modes of SDP

"The knowledge of anything, since all things have causes, is not acquired or complete unless it is known by its causes"

(Avicenna)

MANETs are characterized by high dynamics in particular with respect to the formation of network partitions. This behavior makes it difficult to (s)elect a suitable leader for a given network partition at any given time. However, such leaders or representatives are needed by many applications and functionalities in MANETs. In this chapter, an illustration of the leader election process and its influence on SDP is studied. SDP leaders are assumed to be the popular service providers inside the different network partitions. The presence of higher numbers of active replicas inside the different network partitions has negative effects on optimality of the generated service distributions. The concurrently running active replicas inside one network partition present a group of candidates. Based on the popularity, SDP allow the interested clients in this partition to elect one service provider (leader) and the other should be hibernated. In this chapter, we are motivated to investigate the effects of a set of different leader election mechanisms on the replica allocation process as well as the general service availability, success ratio and prevalence of the protocol.

The leader election problem in MANETs is difficult to address as the topological and resource properties are always changing. Most of the service replication protocols employ proactive leader selection approaches based on partition prediction approaches. Based on different criteria, a leader may be either selected or elected to host a service replica to be operated in the predicted forming partition. When two or more network partitions are being merged, a leader is elected from the concurrent services in the newly formed partition to transform the multiple independent copies of the service into one synchronized service. Most hierarchical routing protocols for MANETs like CEDAR [SSB99] and ZBR [DHY03] address the problem of leader election; mostly resource rich nodes are selected to host information used to produce the required routes by the nodes inside the clusters of these leaders. The TORA [S. 99] routing protocol represented the milestone for the work of [MWV00] to introduce a self stabilizing leader election approach in two modes among single and multiple topological changes. In [DB08b], another self-stabilizing leader election algorithm based on TORA is presented for which it has been proven that it ensures that a network partition can converge to a legitimate state within a finite time even if topological changes occur during the convergence time. [DE08] proposes a hierarchical leader election protocol which clusters the network into balanced clusters, builds a spanning tree of the locally selected cluster leaders, and ends with the election of a super-leader. [VDI+03] considers the problem of trust among the participants in many applications and introduces a set of secure leader election algorithms for the two synchronous and asynchronous modes for MANETs.

In SDP, based only on the amount of interest regarding a specified service , new providers will be chosen to host replicas of this service. In a given network partition, the hibernation mechanism dominates the leader election process. In case of concurrently running replicas, those replicas which are more interesting

from the clients' point of view, will get more requests (interest by the clients in this partition). After a while, the hibernation mechanism will shut down all of the less interesting replicas and only one (a set of replica in some cases, see Chapter 11) replica will continue. This is implicitly the SDP leader election process.
In this chapter, we will enhance the hibernation behavior based on the results of Chapter 7 about the better results of the shorter hibernation test intervals of the hibernation thresholds. Therefore, two leader election mechanisms are proposed. These mechanisms introduce different specifications for the hibernation test interval of the hibernation threshold. The work included in this chapter has been partially published in [HKR09d].

The structure of this chapter is as follows: In Section 8.1 the leader election mechanisms, modes, for SDP are introduced and discussed. Based on a detailed simulation, a evaluation for both of the proposed mechanisms is presented in Section 8.2 where the results have been shown and analyzed. Finally, the work of this chapter is summarized, the contribution are highlighted, and the next research motivations are presented in Section 8.3.

8.1. Leader Election Modes of SDP

At a certain time the service may be replicated over more than one hosting mobile node within the same network partition. This produces a set of equivalent concurrently running replicas. Based on the hibernation mechanism, after a given time interval, the hibernation threshold, and at each of the concurrent provider sites, a check for the gained interest will take place. Since all of the replicas' clients will select the replica with the lowest cost per request in this network partition, just one replica will gain enough interest and will continue. The rest will be hibernated and cached. The main parameter in this selection process, besides the request cost (which steers the common interest to a specific service), is the length of the given time interval to perform the hibernation checks and decisions. This interval is supplied by the hibernation threshold. As mentioned in Chapter 7, the hibernation threshold is formed as *(a number of requests : hibernation time interval)*. The length of the hibernation time interval determines the leader election modes of the service distribution protocol: Longer lengths of the hibernation check interval lead to more simultaneously running replicas inside a given partition and vice versa. The impacts of different settings for this time interval on SDP performance in more details will be addressed and investigated in more detail in the following sections.

The state diagram Figures 8.1 and 8.2, present the proposed election modes of SDP. In these diagrams, regarding a specified service, the states mean:

- A connected client: a client which can achieve a communication to any of the service providers.

- Active service provider: Any mobile host which has an active service or replica and being ready to respond the client requests.
- Service in evaluation: The time that a service provider check the number of gained requests of a specified service.
- Provider with a hibernated service: A mobile node that cached a service or a replica.

On the other hand, the transition conditions contain a set of important constraints like:

- N/A Service provider: No service provider can be reached by a specified mobile host.
- A/V Service provider: At least one provider can be reached by a specified mobile host.
- A request: A client generates a request for a specified service according to its requesting behavior.
- Gained requests: At a provider side, the number of received requests during a certain time interval.
- Cached service: The mobile node has a hibernated service.
- Inactive provider: a mobile with a hibernated service.
- MinTime(publishing-requesting): The minimum required time for publishing a new service or activating a hibernated service and getting at least one service request.

8.1.1. Long-Election Mode of Hibernation

In this mode, the hibernation time interval of the hibernation threshold is fixed and determined to be either one requesting time unit or a multiple thereof. In this work, the requesting time unit is a minute. This mode enables replicas to remain activated for a longer time to be evaluated. We assume the long-election to be helpful for very high mobility networks, in which the frequency of network partition formation is very high. As shown in Figure 8.1, after a fixed evaluation time ends, the service provider evaluates to whether hibernate its replica or not. The election process is done here if only one of the service clients issued a service request to the service before this time interval ends.

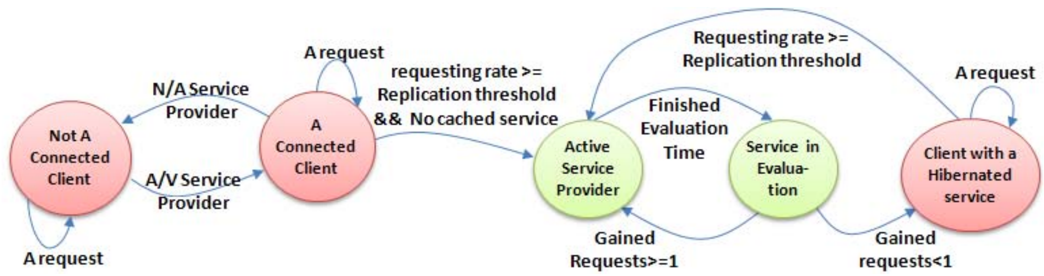


Figure 8.1.: Long-Election mode of the hibernation mechanism

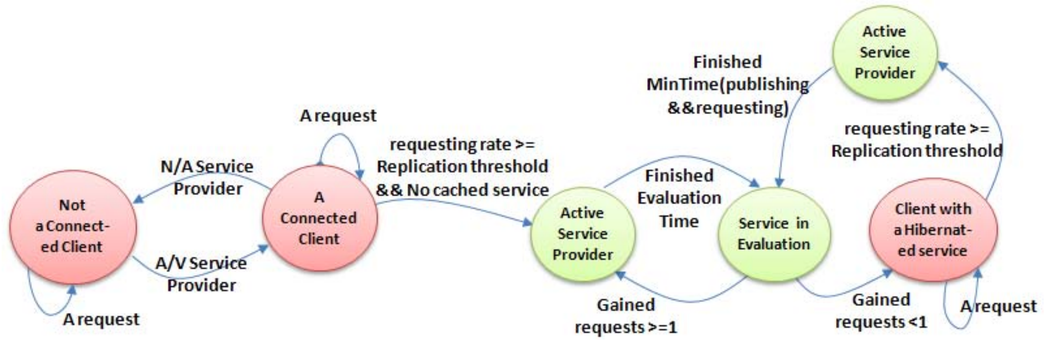


Figure 8.2.: Short-Election Mode of the hibernation mechanism

8.1.2. Short-Election Mode of Hibernation

The basic idea in this mode is to evaluate the initial interest regarding a given (just published) service/replica as soon as possible. This means that evaluation should take place after the minimum possible time required by the service provider to receive, activate, publish, and wait for a slice time for requests. Once this very short time interval ends, the hibernation test takes place. If the service received at least a given number of requests, this means the service will continue and any other active service will be hibernated. This election mode is very conservative regarding the number of concurrent replicas inside a network partition.

In Figure 8.2, SDP evaluates the initial interest regarding a given (just published) service/replica as soon as possible. This means that evaluation will take place after the minimum possible time required by the service provider to receive, activate, publish, and wait for a slice time for requests.

8.2. Evaluation

The evaluation in this section is based on the simulation tool of Appendix A. The general performance has been discussed in terms of service availability, success ratio, prevalence and residence time. Regarding the quality of the replica allocation (placement) process, WRCR (as defined in Chapter 7) and the number of replicas inside the Typical Partition have been investigated. The introduced performance metrics are the same as introduced in Chapter 7.

8.2.1. Configurations and Settings

The simulation run time for each experiment is set to be 2 hours. The network size varies from 10 to 140 mobile hosts. Results come out of the average of 20 times runs. The number of client groups is either 2 groups(maximum requesting rate 1 request/minute) or 4 groups (maximum requesting rate 3 requests/minute). The replication threshold for a client is achieving a number of requests equal to the maximum call rate in a given minute (Max-Requesting-rate:1), the hibernation threshold for a provider is gaining less than one request in a given length of minutes. Regarding the previously introduced two election modes, the hibernation threshold will be either at least one request per minute (1request:1minute, indicates the long election mode) or one request per a minimum waiting time interval shown in Subsections 8.1.1 and 8.1.2. In this simulation, we consider this required time interval to be 1 second.

8.2.2. Results

Figure 8.3 shows the general performance of the service distribution protocol. For both rich and poor interest scenarios, the service availability increases as the network becomes mode dense. This leads to the same behavior on the success ratio since a higher success ratio (indicates the number of succeeded requests in the network) is tightly coupled to a higher service availability. No difference in the results comes from the different election modes. In fact this can be concluded as a positive point for the short election mode: Shorter hibernation check intervals do not affect the high service availability and success ratio. As the number of the client groups increases both the service availability and accordingly the success ratio decreases. This is due to the smaller set of the interested clients in hosting replicas of the client groups. Table 8.1 shows the values of the average service availability and success ratio with the relative observed standard deviation for both scenarios, hibernation modes and number of client groups.

Figure 8.4 shows that the service prevalence and residence time of SDP regarding Scenario 1 and 2 categorized by the number of client groups. As shown in Figure 8.4-

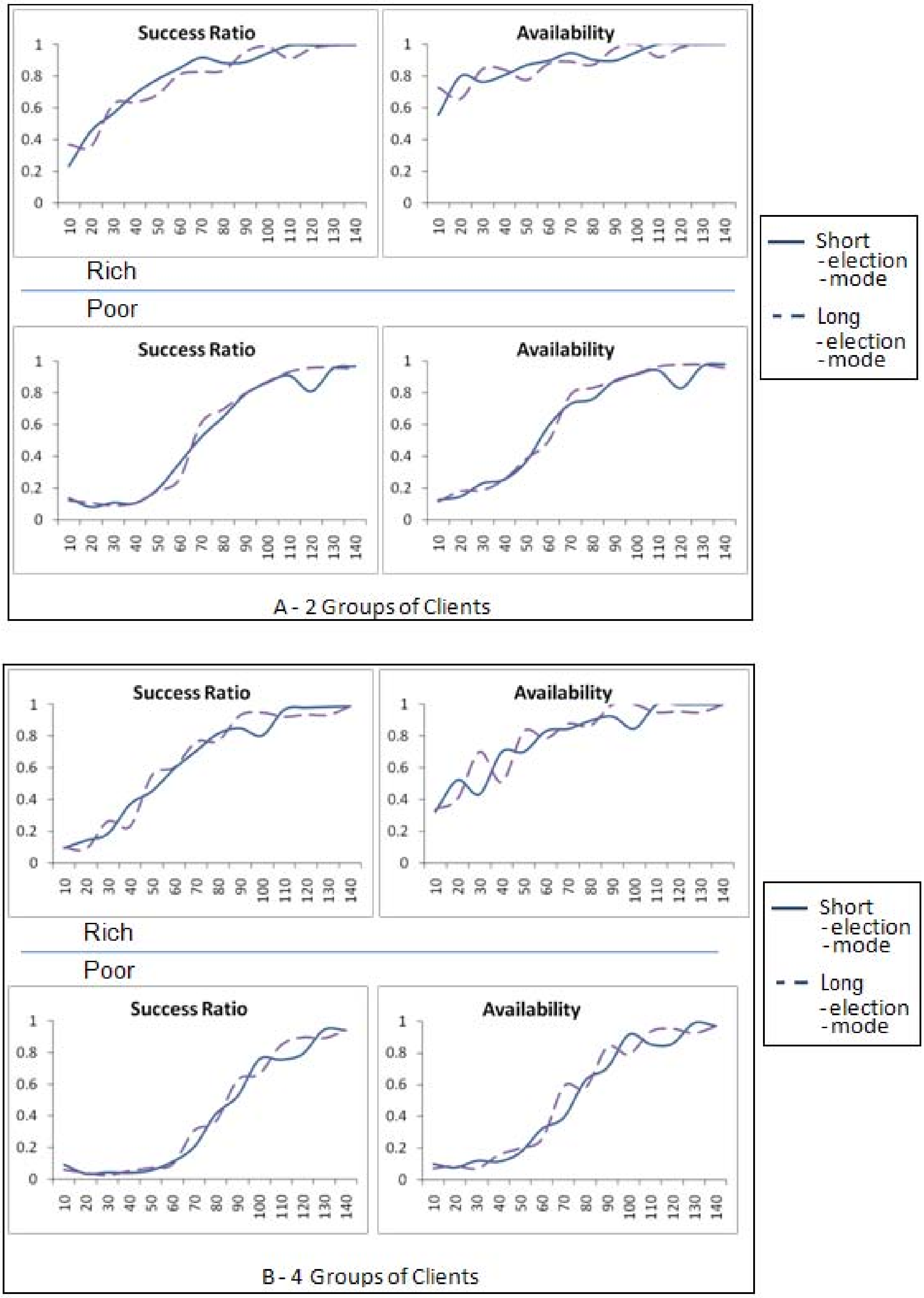


Figure 8.3.: Success ratio and service availability (0-1) against the network size (10-140 node) for both of scenario 1 and 2 for (A) 2 client groups and (B) 4 client groups

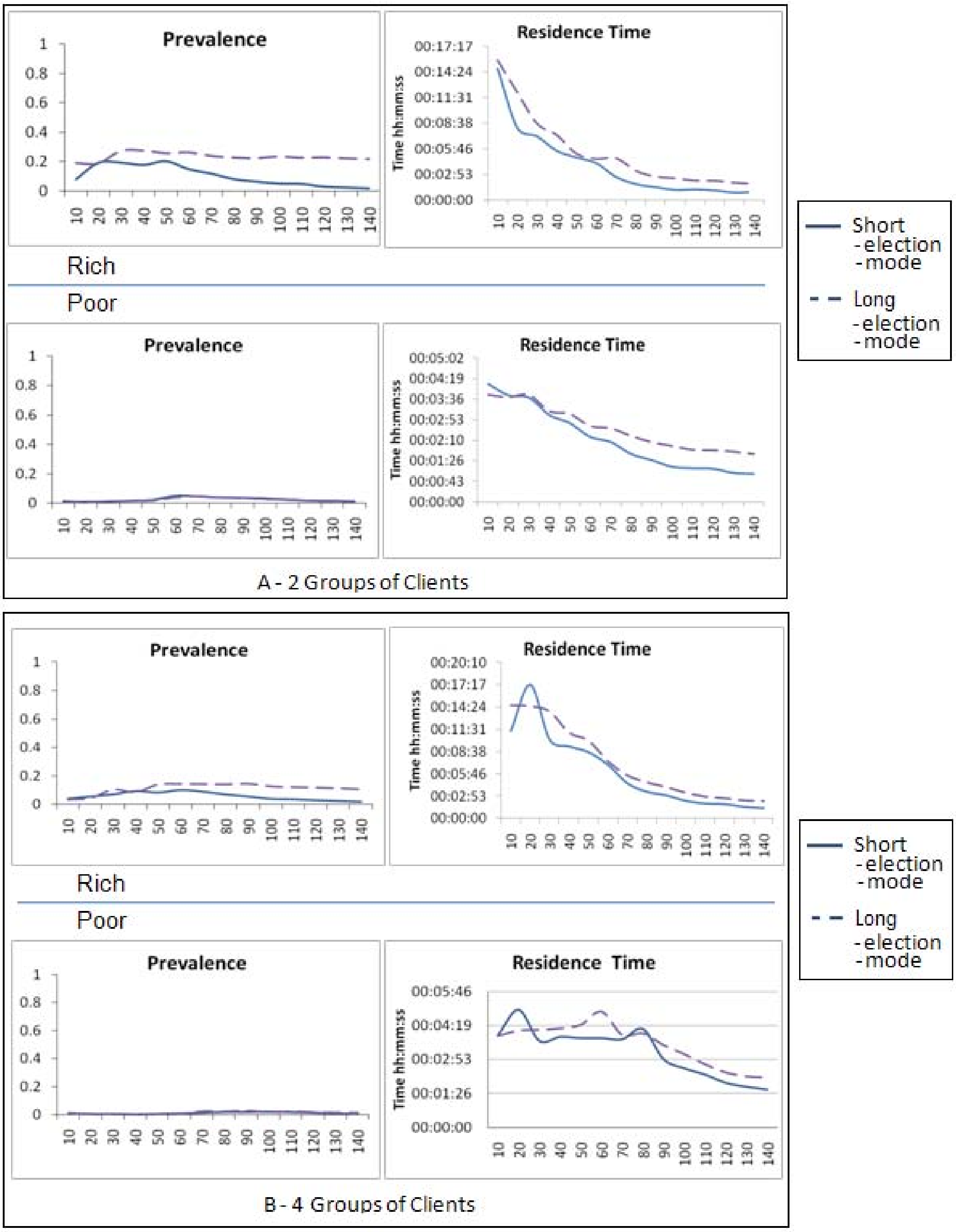


Figure 8.4.: Service prevalence and residence time (0-1) against the network size (10-140 node) for both of scenario 1 and 2 for (A) 2 client groups and (B) 4 client groups

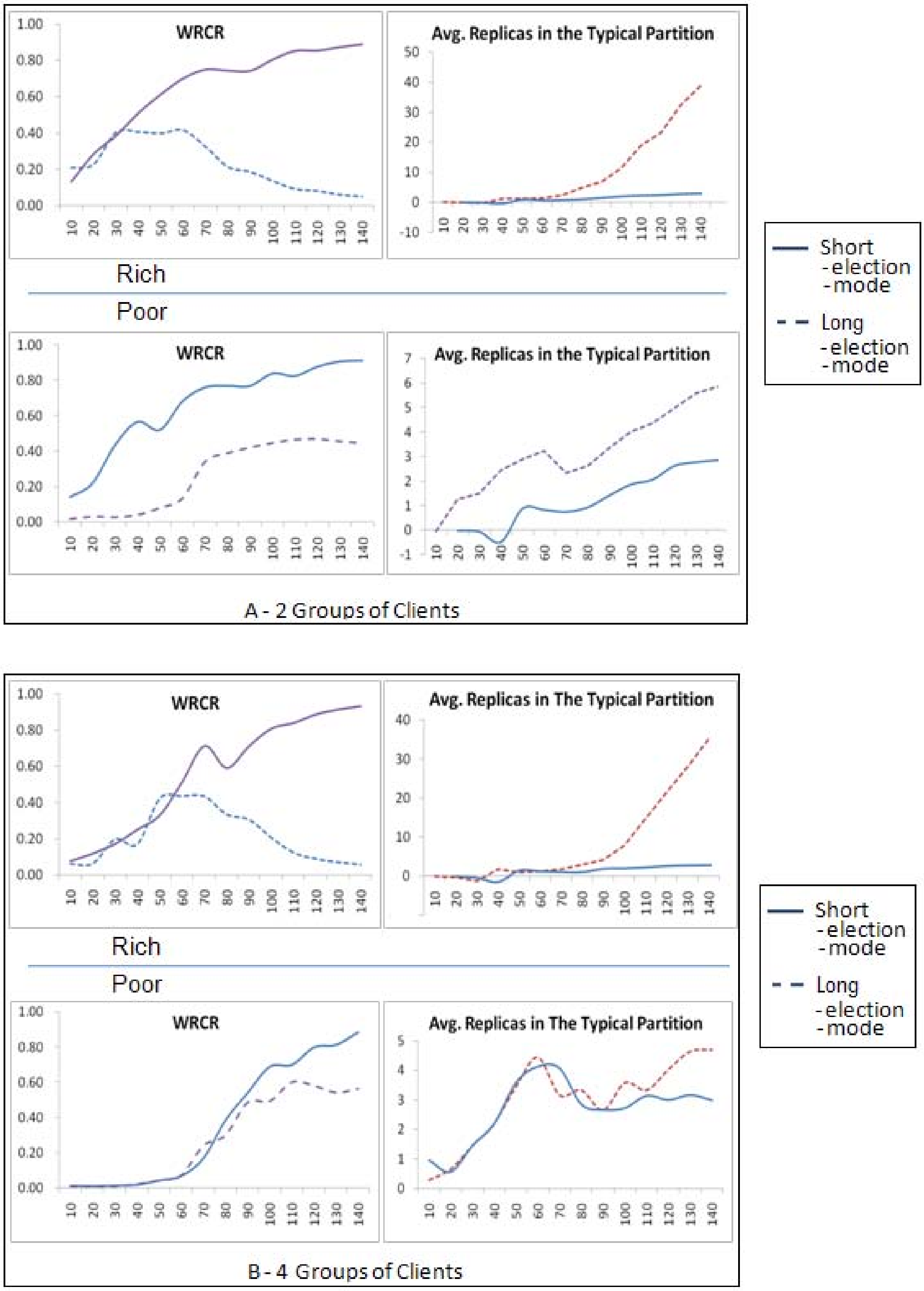


Figure 8.5.: WRCR ratio (0-1) and Avg. number of replicas in the Typical Partition against the network size (10-140 node) for both of scenario 1 and 2 for (A) 2 client groups and (B) 4 client groups

A, in the rich interest scenario (Scenario 1) although the number of the generated requests is large in the two groups of clients, the observed prevalence ratio is about 0.23 in average. The short election mode can even reduce this ratio to be less than 0.10 in average in the higher network sizes (starting from 100 nodes). The decreasing ratio of the service prevalence in this case can be understood from the according curve of the WRCR in Figure 8.5. SDP preserves mostly a fixed number of replicas (relatively optimum) inside each network partition. Therefor, as the network size increases, the service prevalence decreases. On the same manner, the Figure 8.4-B shows the service prevalence but for four groups of clients which means a less number of generated request for the service or its replicas will be found. So, the number of the interested clients will be lower and the service prevalence will decrease (average 0.06). Again, applying the short election mode can reduce the prevalence to be about (average 0.02) especially for the large network sizes. On the other hand, in the experiments where the poor interest scenario is applied in both categories of the number of clients' groups, the number of the generated service requests will be very limited. In these cases, a proper difference between the two proposed election modes was not noticed.
Regarding the service residence time, as shown in Figure 8.4-A and -B, generally, the residence time of both election modes is similar in average. Table 8.2 presents more details about the average values and the standard deviation of the curves of Figure 8.4.

Figure 8.5 shows both the observed WRCR ratio and the average number of concurrent replicas in the Typical Partition in the network regarding varying the network sizes. For Scenario 1 and for 2 client groups, as the network increases, the WRCR ratio increases, too. Starting from 70 node network size WRCR is about/more than 0.80. As the client groups increases, the WRCR ratio growing rate becomes slower. This is due to the lack of service prevalence in the different network partitions. The number of partitions with no service coverages is increased because of the lower number of clients interested to host a replica. The same behavior is presented by the curves of the short election mode for Scenario 1. On the other hand, in the long election mode and for both numbers of client groups, the observed WRCR ratio starts increasing with the increasing network sizes till an interval of network sizes (about [30,60], and [50,70] nodes for (2,4) client groups respectively). Then, the WRCR ratio starts decreasing as the network size increases. This behavior could be interpreted by the curves of the average replicas in theTypical Partition. For 2 client groups, the average number of concurrent running replicas is slightly growing till about network size of 70 nodes (less than 5 replicas in the Typical Partition). Then, it grows quickly to which directly affects the observed WRCR ratio. In the 4 clients groups, the WRCR collapses slower, because the average number of concurrent running replicas is less then in the 2 groups case. In fact, in the high density network (large network sizes), the Typical

Partition size is mostly about the network size, and in case of high Gross Interest and lower replication threshold, as in Scenario 1 with 2 client groups, the number of the interested clients to host replicas is very high and produces a higher number of concurrent replicas. In Scenario 2, in case of poor Gross Interest, the protocol conserves higher WRCR ratio as the network size increases for both of the hibernation modes. Increasing the number of the client groups affects negatively the growth of the WRCR ratio. Generally, applying the short hibernation election mode decreases the number of the concurrently running replicas in all scenarios and client groups (about 3 replicas at most at a time). This result is pretty good because, by applying such an election mode, we can ensure a very limited number of replicas inside the Typical Partition at a time. Table 8.3 presents more details about the average values and the standard deviation of the curves of Figure 8.5.

8.3. Summary

The main contribution in this chapter is to determine how SDP can converge to the optimum replica distribution, in terms of WRCR as mentioned in Section 7.1, based on the proposed leader election modes. Two time based leader election modes, short and long election modes, are introduced for SDP. Results showed that the short election mode can ensure a very limited number of concurrently running replicas from both the whole network and a network partition points of view. Although the long election mode can achieve high performance metrics, regarding service availability and success ratio, it produces a relatively high number of concurrent replicas in the network. On the other hand, the short election mode can also achieve a similar service availability and success ratio a constant number of concurrent replicas and better service prevalence inside the network and its partitions. By the end of this chapter, the questions regarding finding out a proper set of specifications for SDP main core mechanisms of replication and hibernation have been investigated. The effects of different Gross Interest scenarios, replication and hibernation behaviors, and the leader elections mechanisms have been addressed. In the next chapter, the short election mode is assumed as the service selection mechanism, unless something else is stated.

Based on the described SDP core mechanisms, specifications and their influences, and effects of the main factors like service popularity and leader election, the performance of SDP can be expected versus different network constraints.

Further motivation questions:

As discussed in Chapter 5, the mobility behavior of the network participants plays an important role in the service prevalence process. Although we have introduced the concept of the Typical Partition in Chapter 7, the effects of having other mobility models (not Random Waypoint-like) on SDP need to be examined.
Moreover, not only the mobility is a concern but also comparing SDP in terms of performance metrics and replica allocation correctness to others can provide a better image for SDP in action.
The key motivation questions posed at this stage of the research are:

- What are the effects of the other mobility models on SDP performance?
- How is SDP performance compared to others?

These questions and investigations represent the motives behind the next chapter.

			SuccessRatio		Availability	
			Avg.	Avg.StD	Avg.	Avg.StD
Scenario1	2 Groups	Short Mode	0.78	0.20	0.88	0.19
		Long Mode	0.81	0.15	0.90	0.16
	4Groups	Short Mode	0.64	0.17	0.79	0.23
		Long Mode	0.64	0.17	0.80	0.24
Scenario2	2 Groups	Short Mode	0.53	0.14	0.62	0.16
		Long Mode	0.55	0.11	0.64	0.13
	4Groups	Short Mode	0.41	0.13	0.52	0.17
		Long Mode	0.42	0.12	0.53	0.17

Table 8.1.: Average (Avg.) success ratio and service availability with the average value of the observed standard deviation (Avg.Std.) for both Short and long election modes (Short Mode, Long Mode) for both scenarios and client groups

			Prevalence		Residence Time	
			Avg.	Avg.StD	Avg.	Avg.StD
Scenario1	2 Groups	ShortMode	0.10	0.05	00:03:59	00:01:50
		Long Mode	0.23	0.09	00:05:21	00:01:57
	4Groups	Short Mode	0.06	0.04	00:05:57	00:03:21
		Long Mode	0.11	0.05	00:07:05	00:03:41
Scenario2	2 Groups	ShortMode	0.02	0.02	00:02:35	00:00:34
		Long Mode	0.06	0.03	00:04:56	00:00:44
	4Groups	Shor tMode	0.01	0.01	00:03:11	00:01:23
		Long Mode	0.01	0.01	00:03:31	00:01:26

Table 8.2.: Average (Avg.) service prevalence and residence time with the average value of the observed standard deviation (Avg.Std.) for both Short and long election modes (Short Mode, Long Mode) for both scenarios and client groups

			WRCR		Nr.Replicas/Typ.Partition
			Avg.	Avg.StD	Avg.
Scenario1	2 Groups	ShortMode	0.65	0.14	1.41
		Long Mode	0.23	0.11	10.10
	4Groups	Short Mode	0.56	0.16	1.41
		Long Mode	0.21	0.11	8.46
Scenario2	2 Groups	Short Mode	0.66	0.14	1.39
		Long Mode	0.27	0.06	3.18
	4Groups	Short Mode	0.37	0.10	2.68
		Long Mode	0.28	0.09	2.99

Table 8.3.: Average (Avg.) WRCR with the average value of the observed standard deviation (Avg.Std.) and the computed number of replicas inside the Typical Partition (Nr.Replicas/Typ.Partition) for both Short and long election modes (Short Mode, Long Mode) for both scenarios and client groups

CHAPTER 9

SDP In Action

"The constant questioning of our values and achievements is a challenge without which neither science nor society can remain healthy"

(Niels Bohr)

Investigating the application of SDP under more realistic mobility patterns of the network participants and comparisons to competitors represent our objectives in this chapter. In the first half of this chapter, through Sections 9.1 and 9.4, the issues related of applying SDP with a set of realistic mobility models has been addressed. Afterwards, two topology-based service replication protocols have been selected to be compared with SDP. The comparisons have been made to highlight the differences in criteria and performance between SDP as an Interest-based replication protocol and the other two protocols as topology-based protocols. The work of this chapter has been partially published in [HKR10a], [HKR09b] and [HDKR10].

Virtually all service replication approaches for MANETs have been evaluated using simulations mostly based on rather simplistic mobility models, e.g., Random Waypoint model (RWP) [LNR04, LHK05]. In the first part of this chapter, we investigate how big the influence of different realistic mobility models is on SDP performance. We have chosen a variation of the Area Graph Based Mobility Model (AGM) by Bittner et. al [BRS05] as the basis for the experiments. The results are very promising and confirm SDP's previous results.
In order to analyze algorithms and their performance in MANETs, capturing mobility patterns is very important. Real mobility patterns are too hard to be captured. Instead of this, the majority of research in MANETs applies simulations based on synthetic mobility models. RWP mobility model is the most famous mobility model applied by the community of research. RWP is widely used for its simplicity, despite some negative features with spatial mobile node distribution, being memoryless, and sharp turns between its transitions [BHPC04]. A survey [KCC05] on the proceedings of the ACM MobiHoc symposiums between 2000-2005 showed that about 66% of the papers, which used a mobility model in their simulations, stated usage of RWP. In this chapter, a more realistic mobility model, which is AGM [BRS05], is used. In this model, the area which contains the network is divided into places connected through a network of corridors. The idea comes from that the area where the network will be deployed in the real world consists of a set of places connected to each other with a set of ways or corridors. In each place, an arbitrary mobility model may be used. For example, in a university, campus, library, lectures' halls, and cafeteria represent the set of world's places. These places are connected somehow through a set of corridors. Both places and corridors form the area segment set. A complete city can be modeled with its streets as a set of connected places and corridors. In this chapter, a set of different arrangements for the places' topologies is developed. Places' topologies are arranged into a fully connected, ring, lane and star models. Each of these arrangements provide an area graph based-driven mobility model. Moreover, impacts of no and very long corridors have been investigated. By such heterogeneous settings that affect mobility patterns of the network participants, SDP has been verified in more realistic envi-

ronments in terms of mobility.

The work of Section 9.5, comparing SDP to other approaches, was achieved in cooperation with Dr. Abdelouahid Derhab from Department of Computer Engineering, CERIST Center of Research, Algiers, Algeria.
The structure of this chapter is as follows: In Section 9.1, the area graph based mobility model with a set of derived mobility models are introduced and described in detail. A proposed computation for the average velocity is proposed in Section 9.2. Section 9.3 presents the heterogeneity of the proposed general model and specifications scenarios. In Section 9.4, a detailed simulation is introduced. Section 9.5 introduces a simulation based comparison study in terms of performance between SDP and other replication protocols for MANETs with different mobility specifications. Then, the results are presented and discussed. Finally, the work of this chapter is summarized, the contribution are highlighted, and the next research motivations are presented in Section 9.6.

9.1. Area Graph Based Driven Mobility Models and The Network Model

In AGM , and its derivatives in this chapter, a node stays at least for a $MinStayTime$ and no longer than a $MaxStayTime$ inside its current place. Once it decides to leave the current place it randomly selects one desired place to visit. The desired place can not be the current place. Also, the desired place should be directly connected to the current place by a corridor. One and only one corridor is allowed to connect any two places. The selection of the next place to be visited is done according to the weight of the connecting corridor. If the current place is connected to more than one neighboring place, each of the connecting corridors should have a weight and the sum of these weights should be 1. After traversing a connecting corridor, a node enters the desired place which becomes its current place and so on. In each place, nodes are moving using the RWP model. In RWP mobility model, a node moves from its current location to a specified target location with a constant velocity. The speed is uniformly randomly selected between $[0, V_{max}]$. After reaching its destination, it waits till a pause time ends. The pause time is uniformly selected from $[0, MaxPaueTime]$, then it finds another target location to go and so on. Each of the mobile nodes can cover a circular transmission range around itself with a fixed radius R in places or corridors. If the distance between two mobile nodes is less than or equal R, a wireless link between them will be established. Fully connected places, ring, star, and lane places' arrangements represent the proposed different area graph based mobility model derivatives. Each of them is considered as a mobility model.

9.1.1. Network Model for Fully Connected Arrangement of Places

The network model for AGM is proposed based on the extended network model of Chapter 5 in which the set of mobile nodes N with a fixed network size $NetworkSize$, $N = \{n_0, n_1, ..., n_{NetworkSize-1}\}$, forms the network participants. Due to the motion of mobile nodes, a set of edges (links) E between them is produced. An undirected graph G can model the network as a union of non overlapped graphs, network partitions, such that:

$$G(N, E) = G_1(N_1, E_1) \cup ...G_x(N_x, E_x) \cup ...G_k(N_k, E_k) \tag{9.1}$$

where:

$$N = N_1 \cup ... \cup N_x \cup ... \cup N_k,$$
$$E = E_1 \cup ... \cup E_x \cup ... \cup E_k,$$
$G_x(N_x, E_x)$ represents the x^{th} network partition.

$G(N, E)$ is put overall the area's places and corridors (area segments). The area segments (S) can be defined as :

$$S = P \cup C \tag{9.2}$$

where (P) is the set of places:

$$P = \{P_1, P_2, ..., P_i, ..., P_{max}\} \tag{9.3}$$

and (C) is the set of corridors:

$$C = \{c_{ij} | i, j \in P, i \neq j, c_{ij} \equiv c_{ji}\} \tag{9.4}$$

Let uss define a spatial function, $SpPos(n_i)$, which returns the spatial coordinates of a given mobile node n_i, for simplicity (x,y) coordinates are used, where:

$$SpPos(n_i) = (x_{n_i}, y_{n_i}) \tag{9.5}$$

We define a spatial function, $SpContained((x, y), S_i)$, which indicates if a specified position in (x, y) coordinates is spatially contained by a given area segment S, where:

$$SpContained((x, y), S_i) = \begin{cases} 1 & (x, y) \in (S_i) \\ 0 & otherwise \end{cases} \tag{9.6}$$

We define a spatial function, $SpIn(n_i, S_i)$, which determines whether a mobile node n_i spatial position is in a given segment S_i where:

$$SpIn(n_i, S_i) = \begin{cases} 1 & SpContained(SpPos(n_i), S_i) = 1 \\ 0 & otherwise \end{cases} \tag{9.7}$$

and we define $Contains(S_x, G_x)$ which indicates if a network partition G_x is contained by S_x, where:

$$Contains(S_x, G_x) = \begin{cases} 1 & \forall n_x \in G_x(N_x, E_x), SpIn(n_x, S_x) = 1 \\ 0 & otherwise \end{cases} \tag{9.8}$$

At a certain time t: $\forall G_x \in G\ \exists$ one (and only one) $S_x \in S$ such that $Contains(S_x, G_x) = 1$ and $Contains(S - S_x, G_x) = 0$, which means that the network partition could not be formed over more than one area segment. A link is formed between two mobile nodes n_i, n_j if both of them is in the same area segment and the distance between them is less than a specified radio transmission range R as follows:

$$link(n_i, n_j) = \begin{cases} 1 & Ecld(SpPos(n_i), SpPos(n_j)) \leq R \\ & , SpIn(n_i, S_x) = SpIn(n_j, S_x) = 1 \\ & , i \neq j \\ 0 & otherwise \end{cases} \tag{9.9}$$

where $Ecld(Pos_a, Pos_b)$ is a function computes the Euclidean distance between two nodes. By using the previous equation, we can specify the set of edges E in equation 9.1 to be:

$$E = \{e_{ij} | i, j \in N, link(i, j) = 1, i \neq j\} \tag{9.10}$$

9.1.2. Network Model for Ring Arrangement of Places

We define a function, $neighbors_R(P_i)$, to indicate the set of allowed neighbor places to be connected to P_i by corridors, where:

$$neighbors_R(P_i) = \begin{cases} \{P_{i+1}, P_{i-1}\} & Networksize - 1 < i < 1 \\ \{P_i, P_{max}\} & i = 0 \\ \{P_1, P_{i-1}\} & i = Networksize - 1 \end{cases} \tag{9.11}$$

The set of corridors for this arrangement will be C_R, where:

$$C_R = \{c_{ij} | i, j \in P, j \in neighbors_R(i), c_{ij} \equiv c_{ji}\} \tag{9.12}$$

9.1.3. Network Model for Lane Arrangement of Places

This arrangement requires modifying the neighbor function in Equation 9.11 to be as follows:

$$neighbors_T(P_i) = \begin{cases} \{P_{i+1}, P_{i-1}\} & Networksize - 1 < i < 1 \\ \{P_i\} & i = 0 \\ \{P_{i-1}\} & i = Networksize - 1 \end{cases} \tag{9.13}$$

So the set of corridors C_T will be:

$$C_T = \{c_{ij} | i, j \in P, j \in neighbors_T(i), c_{ij} \equiv c_{ji}\} \tag{9.14}$$

9.1.4. Network Model for Star Arrangement of Places

In this arrangement, all the places are connected just to a root place. For simplicity, the first place P_1 is considered as the root place. So, the set of corridors in this case C_S will be:

$$C_S = \{c_{iP_1} | i, P_1 \in P, i \neq P_1, c_{iP_1} \equiv c_{P_1 i}\} \tag{9.15}$$

9.1.5. General Settings

The specifications of the places in their different arrangements and the related properties of the contained mobile nodes are set as follows:

- Four places are used, each is a $300m \times 300m$ square.
- Radius of transmission range R is 75 meters.
- the default length of the corridors is 100 meters.
- $MinStayTime$ for all places is 5 minutes.
- $MaxStayTime$ for all places is 30 minutes.
- In each place, the RWP mobility model is applied. For the sake of heterogeneity, different values are assigned for both V_{max} and $MaxPaueTime$ as discussed later.
- For simplicity, the outgoing corridors of a place have the same selection weights.

9.2. Average Velocity

We want to find a single measure to capture the different settings of RWP in the given places in the proposed area graph based driven mobility models. As mentioned in [BHPC04], it is too hard to attribute all the features of RWP in terms of a single given measurement. Since, the mobility behavior under RWP is not stable because of the infinite residence slots that a mobile node may have, if it has a zero speed, the average velocity at certain time is introduced (V_{avg}) to reflect the mobility behavior independent of the resident nodes.

RWP can be considered as follows: it consists of a set of consecutive cycles. In each cycle, a node moves for a set of time slots, then becomes resident for another set of time slots. In our RWP settings, the expected number of time slots that the node is being resident can be easily computed. The challenge comes from the fact that the number of expected time slots for the moving interval in a given cycle is hard to capture because it depends on the network area dimensions, shape of

the area and the selected speed of the node. In [BHPC04], a set of measurements that highlight the behavior of mobile nodes under the influence of RWP in terms of nodes' spatial distribution, movement directions, and speed are presented. Based on an important result of $E\{L\}$ (where $E\{L\}$ is the expected next transition length of the next movement in an area of size $a \times a$), $E\{L\} = 0.5214a$. The number of expected time slots n_t given for a node to move is driven as follows:
$n_t = \frac{E\{L\}}{V_{exp}}$, where V_{exp} is the expected velocity of the node. Since, in our RWP settings, $V \in [0, V_{max}]$, $V_{exp} = 0.5V_{max}$

$$n_t = \frac{E\{L\}}{0.5V_{max}} \tag{9.16}$$

Let m_t be the maximum time slots for a node to be residence in a place.
Let $Prob(moving)$ be the probability for a node of being moving, where:

$$Prob(moving) = \frac{n_t}{n_t + m_t} \tag{9.17}$$

and so we can compute the expected number of moving nodes at given time t, $E_t(moving)$.

$$E_t(moving) = NetworkSize \times \frac{n_t}{n_t + m_t} \tag{9.18}$$

The same thing for computing the expected number of pausing (resident) nodes $E_t(pausing)$ at given time t

$$E_t(pausing) = NetworkSize \times \frac{m_t}{n_t + m_t} \tag{9.19}$$

Let $v(i)$ be the current velocity by the i^{th} node at time t. let $Sum(V)$ be the summation of velocities of all networks nodes such that:
$Sum(V) = \sum_{i}^{E_t(moving)} v(i) + \sum_{i}^{E_t(pausing)} v(i)$. Since $v(i) = 0$ for any i^{th} resident node, term $\sum_{i}^{E_t(pausing)} v(i) = 0$, therefore:

$$Sum(V) = \sum_{i}^{\lceil E_t(moving) \rceil} v(i) \tag{9.20}$$

Let us define V_{avg} which is the average velocity of all mobile nodes in a given network at certain time as $V_{avg} = NetworkSize^{-1} \times Sum(V)$, given $E_{v(i)}$ which is the expected value of $v(i)$. Since $v(i) \in [0, V_{max}]$ is uniformly selected, then $E_{v(i)} = 0.5V_{max}$ and by simplifying the previously mentioned equations, V_{avg} can be defined as follows:

$$V_{avg} = E\{L\} \times (2 \times \frac{E\{L\}}{V_{max}} + Ex_m)^{-1} \tag{9.21}$$

	Place 1	Place 2	Place 3	Place 4
Length(m)	300	300	300	300
Width(m)	300	300	300	300
Min.Time(mn.)	5	5	5	5
Max.Time(mn.)	30	30	30	30
Vmax(m/s)	3	6	3	6
MaxPauseTime(mn.)	3	3	15	15
GrossInerest	Rich	Poor	Rich	Poor
V_{avg} (m/s)	0.81	1.1	0.28	0.31

Table 9.1.: Different settings for places, mobility and the assigned Gross Interest

The previous equation is dimensionally correct and independent of the network lifetime. V_{avg} can describe the mobility of a set of mobile nodes at a certain time t. More expressive sophisticated measurements that can attribute RWP mobility model , which is applied in each of the different four places but with different settings, are given in Table 9.1.

9.3. Heterogeneity of The Network Model

The network model proposes a set of four place arrangements. In each place, a unique set of different settings of mobility and requesting behavior is proposed and we assume this is very realistic. For example, imagine a gateway service that facilitates Internet access for the other network participants. Also, assume the different places are the university library, campus, lecture hall, and cafeteria. Both mobility settings for mobile nodes and Internet access behavior should be very different in each of these places. So, Table 9.1 shows the heterogeneous more realistic settings of the four places.

The impact of low and high average velocities on different gross interests are represented. Places 1 and 2 differ in their Gross Interest as they have high average velocities. While Places 3 and 4 differ in the applied Gross Interest scenarios (see Chapter 6), they have low average velocities.

9.4. Evaluation

Based on the simulation tools of Appendix C, a detailed simulation has been introduced for the evaluation of SDP in the proposed mobility models. The simulation time has been set to be 4 hours. The maximum requesting rate is set to 3 requests per minute which means there exist 4 groups of clients (0 requests/minute,

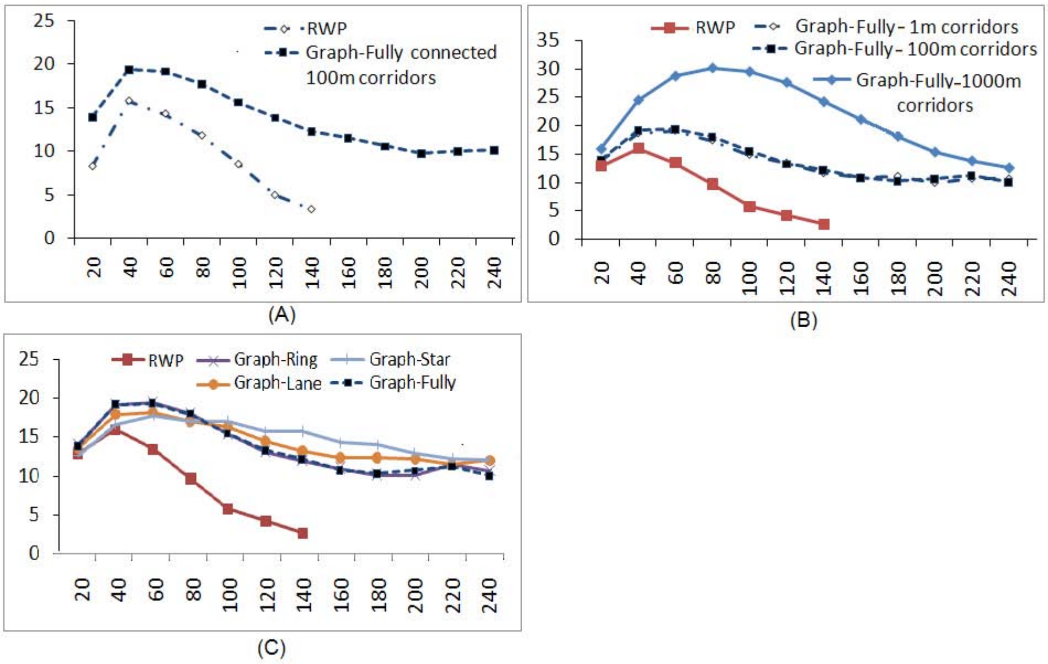


Figure 9.1.: Number of the formed network partitions vs. the network size

1 requests/minute, ..., 3 requests/minute). The replication threshold is set to be 3 requests: 1minute which means that any client can achieve this rate at certain time will ask to host a replica.
As shown in Chapter 8, the hibernation threshold mode is short election mode with one second for the publishing process. Otherwise, the hibernation minimum time interval is one minute. The Requirements' Index (see Chapter 6) varies its 20% normally about a general index of 0.5.

9.4.1. Performance Parameters

Based on the definition of the performance metrics in Chapter 4, 5 and 7, the performance metrics in this evaluation are:

- Service Availability
- Success ratio
- Prevalence
- Residence time

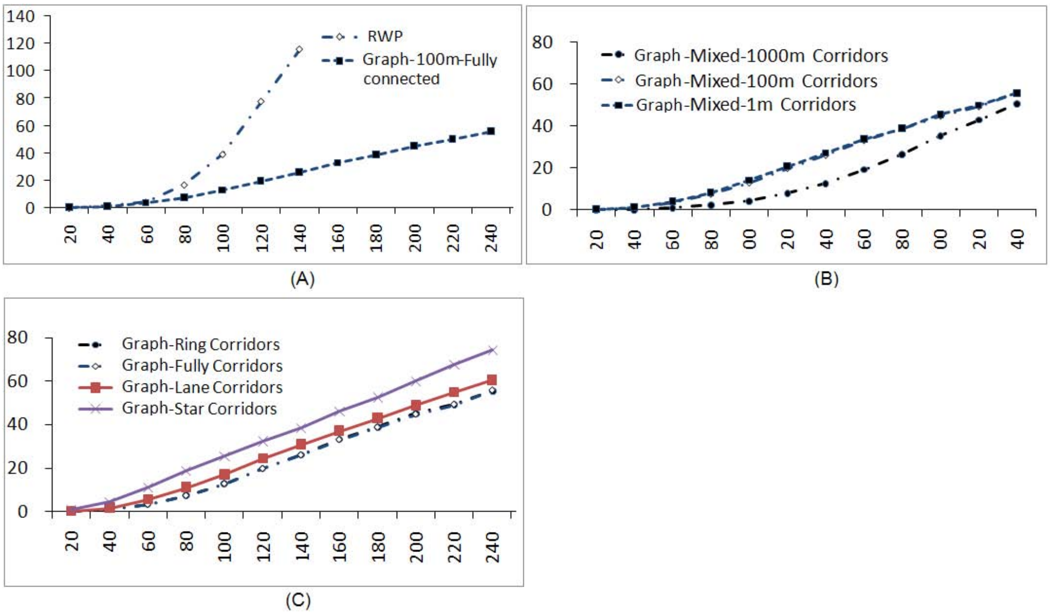


Figure 9.2.: Size of the Typical Partition (Typ.Part.) vs. the network size

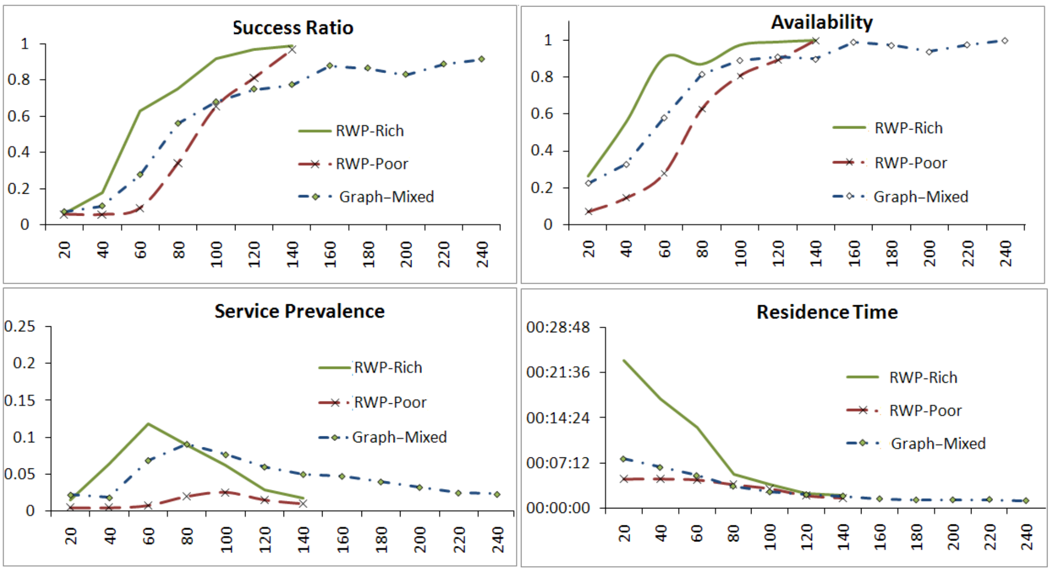


Figure 9.3.: SDP general performance parameters vs. the network size

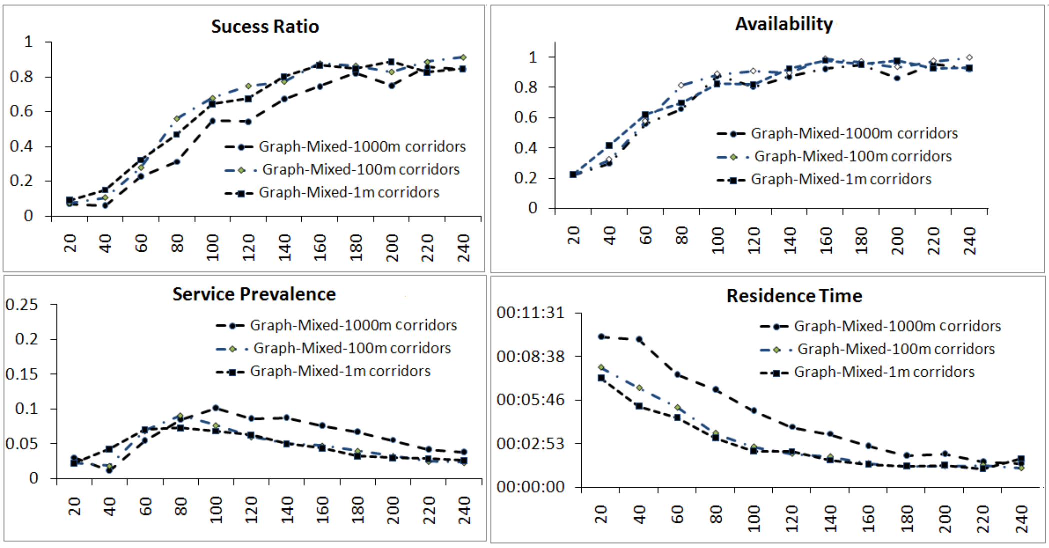


Figure 9.4.: SDP general performance parameters vs. the network size (different corridors' lengths)

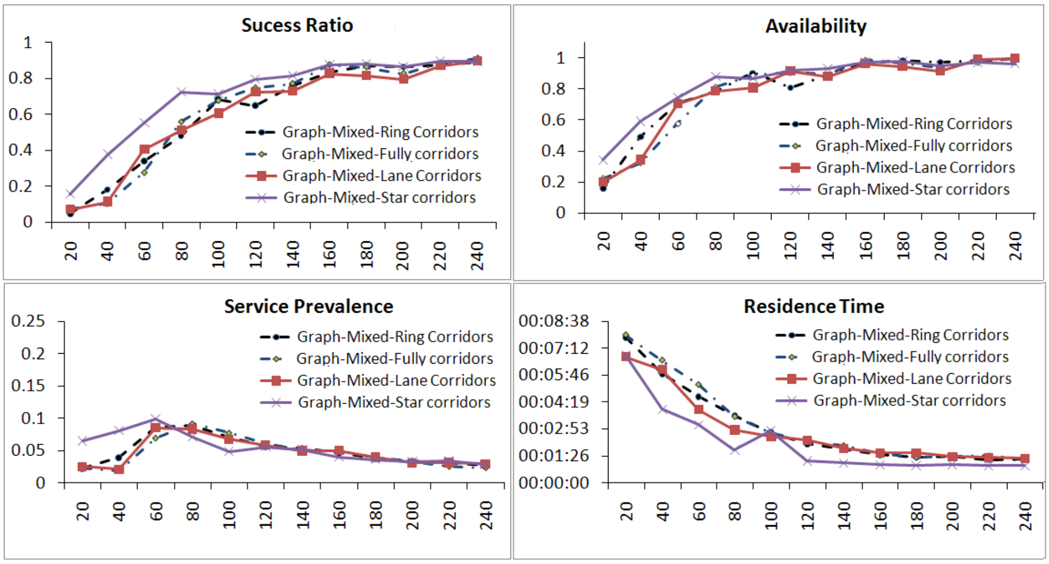


Figure 9.5.: SDP general performance parameters vs. the network size (different arrangements for the area segments)

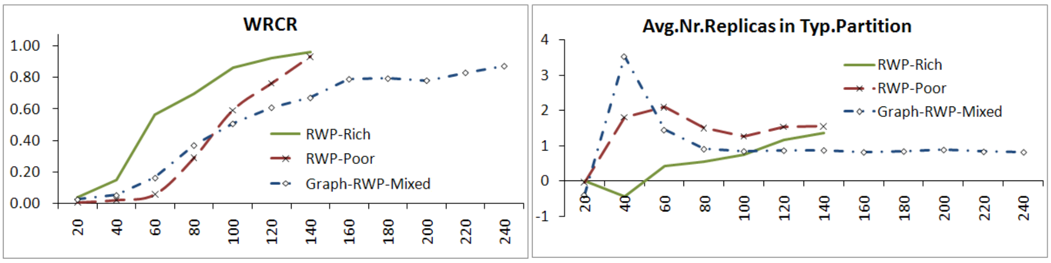


Figure 9.6.: WRCR and the average number of replicas inside Typ.Partition. (Avg.Nr.Replicas in Typ.Partition)

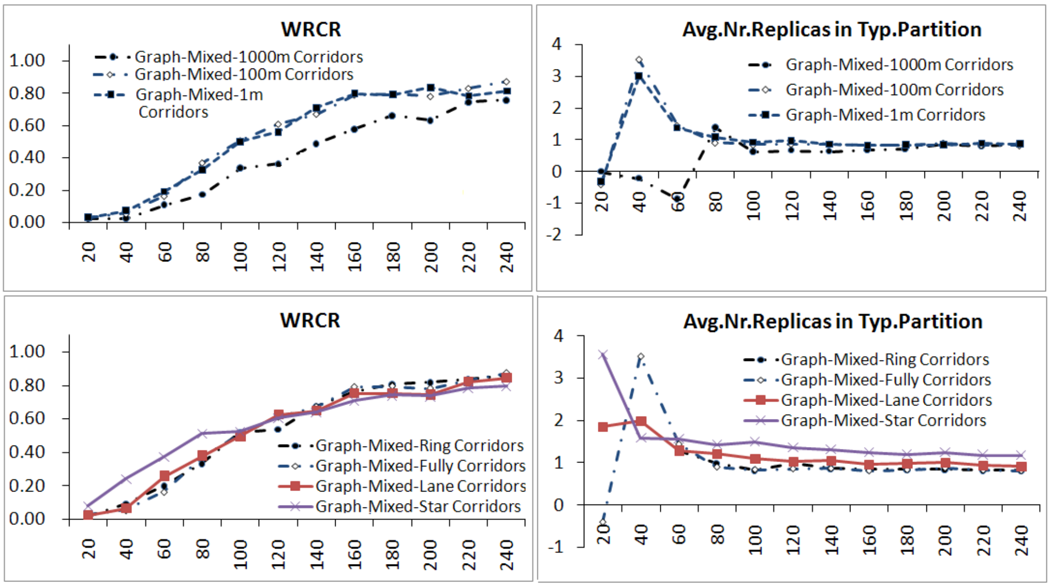


Figure 9.7.: SDP replica allocation correctness (WRCR) and average number of replicas inside the Typical Partition (Avg.Nr.Replicas in Typ.Partition) for both of different corridors' lengths and arrangements for the area segments

- Replica Allocation Correctness Ratio (WRCR)
- Typical Partition (Typ.Part.)

9.4.2. Performance Analysis

Let us denote results which are coming from applying SDP in a $600m \times 600m$ place as “RWP”- group and those coming from applying SDP with different graph based mobility models “Graph”- group. We may add after the group name either a Gross Interest indicator (like “Rich”, “Poor”, and “Mixed”), a corridor length indicator (like “1m”, “100m”, and “1000m corridors”), or a topology indicator like (“Star”, “Lane”, “Ring”, and “Fully connected”).
Since the partitioning behavior of MANETs represents the main motive for service replication, understanding this behavior is very important because, as we are going to see, it can provide a good reasoning for the further results.
Figure 9.1-A shows a significant difference between the RWP group and the Graph group with fully connected places and 100 m corridors. As the network size increases the average number of partitions increases then it is reflected and starts decreasing. The RWP curve is close to the 1 partition starting from network size of 140 nodes. On the other hand, the Graph curves degrade slower. It becomes stable at about 10 partitions starting from network size of 160 nodes. In Figure 9.1-B, the growing in the number of partitions is slower for both lane and star places. This can be due to the fact that the nodes are be more concentrated in either middle places (in the lane arrangement) or the center places (in the star arrangement). After a certain interval of network sizes, the curves start degradation in the different place arrangements. Figure 9.1-C shows the corridor length effects on the average number of partitions. The very long 1000m corridors provide a higher number of partitions. The generated number of partitions for both 1m and 100m are identical. This is due to the node speed in corridors. It has been set to be at least 1 m/s which means any node will traverse the corridor of 1m in 1s and the 100m corridor for 100s which is relatively short compared to the total simulation time.
Not only the number of partitions is an important measurement that can affect SDP performance but also the size of the partition. The Typical Partition can provide a good sample for the most weighted sized network partition. Figure 9.2 gives a view on the Typical Partition through the different experiments. Figure 9.2-A shows that, in RWP, as the network size increases the nodes are mostly located in the same network partition. Starting from network size of 140 node the Typical Partition is about 125 nodes, which means no need to increase the network sizes in our experiments for the RWP group. On the other hand, the corridor length has negative and inverse effects on the size of the Typical Partition as in Figure 9.2-B. The longer corridor length the smaller size of typical partitions because more nodes

will consume time in passing these corridors. Since the corridors are very long, the probability of having higher partitions' sizes inside the corridors is increased. Generally the size of the produced typical partitions are relatively low in all of graph-based mobility models. In 9.2-C, the size of the Typical Partition for both of the lane and star models are higher as they concentrate more nodes in the inner places (either center place in the star arrangement or the middle places in the lane one). Generally, we can conclude that SDP in our simulation is facing at least about 10 network partitions with sizes at most less than 25% of the total network size for all the proposed graph-based driven mobility models. The standard deviation for both number of partitions and size of the Typical Partition is equal 1.5 and 0.7 (for the Graph group) respectively.

Effects of Mixed Gross Interest

As previously mentioned, each of the four places in our models enforces its contained nodes to obey a certain Gross Interest scenario (see Table 9.1). The resultant general Gross Interest in the four given places is denoted the "Mixed" Gross Interest. A comparison between RWP group (which applies either "Poor" and "Rich" Gross Interest individually) and the Graph group with a mixed Gross Interest is presented. Differences between the more-popular/less-popular services (rich/poor Gross Interest scenarios) SDP settings for replication and the proposed (more realistic) "Mixed" applied Gross Interest are investigated. Figure 9.3 shows the general SDP performance parameters (service availability, success ratioo, prevalence andresidence time) and how the proposed mixed Gross Interest (of the graph group) is relatively tensing more to the rich Gross Interest (of the RWP group). Starting from 140 nodes(about 11 network partitions (with standard deviation 0.9)), the Typical Partition size is equal to 45 nodes in average (with standard deviation 2). Average values of service availability, success ratio, prevalence andresidence time (average, standard deviation) are respectively as follows (0.96, 0.01), (0.85, 0.01), (0.04, 0.02), and (00:01:31, 00:00:49) hh:mm:ss. Applying a mixed Gross Interest has relatively low impacts on SDP general performance parameters in despite of presence of large number of partitions and relatively low sizes of typical partitions.

Effects of Variant Corridor Lengths

Length of the connecting corridors between the four places in our model affects clearly the general SDP performance parameters. This experiment is applied only on the fully connected arrangement of places. On the one hand, in Figure 9.4, both curves of service availability, success ratio, prevalence and residence time for 1m and 100m corridors are approximately similar. According to the previously mentioned partition analysis, this is due to the similar number of both formed network partitions and size of the Typical Partition. On the other hand, generally SDP performance parameters under the influence of the 1000m corridor length need higher network densities to reach a performance like the shorter corridor lengths, particularly in both prevalence and residence time. Starting from network size of 140 nodes, for Graph-Mixed-1m, Graph-Mixed-100m and Graph-Mixed-1000m groups respectively as (average, standard deviation), service availability values are (0.84, 0.14), (0.86, 0.09) and (0.78, 0.15), values of success ratio are (0.95, 0.14), (0.96, 0.09) and (0.92, 0.16), values of prevalence ratio are (0.04, 0.01), (0.04, 0.01) and (0.06, 0.02) and values of residence time (hh:mm:ss) are (00:01:33, 00:00:38), (00:01:31, 00:00:25) and (00:02:18, 00:00:38). Since both prevalence and residence time can provide indication of utilization of the network resources, the previous results show how the long corridors affect negatively the network resources. SDP proved in this experiment that it can stand with a huge number of partitions, in cases of Graph-Mixed-1000m, with a stable performance.

Effects of Different Arrangement for The Places

Fully connected, ring, lane and star arrangements for places have been introduced by the proposed heterogeneous mobility models. In this part of our analysis we fix the corridors' lengths to be 100m. Figure 9.5 shows the SDP performance parameters against these different place arrangements. Regarding the partitioning behavior of the different place arrangements, lane and star arrangements have less number of network partitions and more dense Typical Partition. This leads these two arrangements to have better performance than the others. The values of Graph-Mixed-Ring, Graph-Mixed-Fully, Graph-Mixed-Lane and Graph-Mixed-Star groups respectively as (average, standard deviation) are as follows: service availability values are (0.81, 0.15), (0.79, 0.17), (0.79, 0.18) and (0.84, 0.17), success ratio are (0.62, 0.12), (0.63, 0.13), (0.62, 0.14) and (0.71, 0.14), prevalence ratio is (0.05, 0.02) for all arrangements. Finally, the residence time values are (00:02:57, 00:00:52), (00:03:07, 00:01:13), (00:02:47, 00:00:56) and (00:02:07, 00:01:16). Although the middle places of the star and lane arrangements have more nodes, only the star arrangement has a better SDP performance measurements because the mobile nodes are forced to visit the root place from the other places once they leave their current places. In the lane arrangement, probability of visiting another inner place is less than the same probability in the star arrangement. Generally, SDP showed in this experiment that it can preserve its performance even independent of the different arrangements of places.

Impacts on Replica Allocation Correctness Ratio (WRCR) and The Average Number of Replicas Inside The Typical Partition

Figure 9.6 shows (relative to RWP-Rich and RWP-Poor groups) the resultant WRCR of the Graph-Mixed-Fully connected group. As the number of network partitions increases, WRCR increases. For example, at 240 nodes, it becomes (in terms of average,standard deviation) (0.74, 0.17). This ratio is reached at size of Typical Partition equals to, (average,standard deviation), (60, 0.9). The relative low WRCRs, compared to the achieved ratios in both RWP-Rich and RWP-Poor groups, are due to the fact of the low size of the Typical Partition. However, the average number of replicas inside the Typical Partition reflects that the replica distribution inside the network partitions is close to the optimum assumed replica distribution.
In Figure 9.7, regarding different arrangements of places, star and lane arrangements achieve a higher growth in WRCR than the fully and ring arrangements. From the number of replicas per Typical Partition perspective, the higher the number of connecting corridors (like in the fully and ring arrangements) the nearer the results to the optimum case of replication. The final conclusions of the first half of this chapter is going to be mentioned in Section 9.6.

In general, all of MANETs' service replication protocols, introduced in Chapter 3, share a main feature: they query the lower network layer, e.g., the routing components, to obtain information. Besides being expensive and time consuming, these querying processes make the proposed protocols and algorithms architecture-dependent. In the second half of this chapter, we aim to compare the performance of topology prediction based protocols to SDP. Based on the various concepts behind service replication in MANETs and how the existing protocols and algorithms achieve these concepts and the detailed comparison of their features in Chapter 3, the performance of SDP compared to two other replication protocols, namely: (a) The pull-based service replication protocol for mobile networks (PSRP) [DB07] and (b) A Self-stabilizing PSRP (SSRP) [DB08a]. Through the next section, we present this comparison and state out conclusions about the gained results.

9.5. Service Replication Approaches: Interest versus Topology Prediction

PSRP and SSRP represent typical traditional partition prediction based service replication protocols for MANETs. As previously discussed in Chapter 3, they introduced a complete partition detection algorithm based on link evaluation features with the help of a well described routing algorithm (TORA). Regardless of the issue of QoS requirements, PSRP and SSRP assume as SDP that one active service or replica inside a network partition can satisfy all of these partition participants' requests. The performance matrices used in this case study are merged and unified from the introduced matrices in [HDKR10, DB07, DB08a], they can be summarized as follows:

- General Service Availability (for PSRP and SSRP): it is the ratio between the number of nodes that can access the service to the network size (total number of nodes). The introduced general service availability here is differing from the SDP service availability which has been introduced in Chapter 3. While SDP service availability measures the service readiness for replication (the time that at least one service or replica was ready to be replicated to the whole network operation time), the general service availability measures the service accessibility from clients' perspective.
- Success ratio (for SDP)
  SDP distinguishes between services that are important to the clients and the other unimportant services. SDP therefore tries to maximize the availability of the important services and does not care about the others. Therefore, using the standard service availability metrics would be unfair towards SDP. On the other hand, using SDP specific criteria would be meaningless for PSRP and SSRP that have no notion of interest in a service. However, from a user's perspective the semantics behind those two measurements is the same: Both measures how likely he is to get access to a service he wants to use.
  Therefore, it is more convenient to compare PSRPs and SSRPs service availability to SDPs success ratio.
- Weighted Rational Allocation Correctness Ratio (WRCR) as defined in Chapter 7.

In order to understand a comparison between SDP and both of PSRP and SSRP, the following facts need to be considered.

1. For SDP, it is important to have a requesting model. This is the main influence in the replication/hibernation processes. It determines the generated requests and how the clients produce them regarding the service content. In this section, the proposed two gross interest scenarios of Chapter 6 are dominated by the requesting behavior of the clients for SDP. Two different client groups (2 and 4 client groups categories) are used. These groups differ in their maximum requesting rate. In cases of 2 Client groups, the maximum requesting rate will be 1 request/minute (client groups: 0 and 1 requests/minute). In cases of 4 Client groups, the maximum requesting rate will be

1 request/minute (client groups: 0,1, ... 3 requests/minute). The replication threshold, i.e., the minimum requesting rate a client must reach in order to be allowed to obtain a replica (set to equal the maximum requesting rate). Regarding the replication threshold, the higher the number of client groups the lower number of clients ready to receive their own replicas we get. Regarding the gross interest, the richer the gross interest the higher the number of clients ready to receive their own replicas we get.

2. PSRP and SSRP assume that all of the deployed services are important to be replicated once they detected motive changes in the network status or topology. So, we can note that the services to be replicated using these protocols are gaining, in terms of SDP gross interest, a very rich gross interest. In spite of this note, in this comparison, results are also investigated using a poor gross interest settings for SDP. These results, as we are going to show, represent also the performance of SDP with uninteresting services and should be considered in the advantages for SDP.

9.5.1. Settings

SDP settings: A short election mode (see Chapter 8) for the concurrent replicas is used by SDP with one second time interval for the publishing process. Otherwise, the hibernation minimum time interval is one minute. Finally, a general requirements' index (see Chapter 6) is attributing the different services and replicas which differs a 20% normally about a general index of 0.5.

Network settings: RWP as a mobility model [BHPC04] is applied, in which each node selects uniformly and randomly a destination in a $600m \times 600m$ square area with no obstacles and a speed between 0 and a certain maximum speed V_{max} (6 m/s), then it stays during a pause time. Based on the pause time specifications, two mobility groups are introduced:

- "highMobility" where the pause time is between 0 and 15 seconds.
- "lowMobility" where the pause time is between 0 and 15 minutes.

The two mobility groups indicate the performance of the three protocols in different mobility behavior of the network participants. In the "lowMobility" group, the network partitions are formed in an infrequent and slower way than the "highMobility". After the pause time is finished, a new random destination and a speed are selected and so on. Each mobile node covers a 75 m radio transmission range. One node is selected to host the original service in the beginning of the simulation. The simulation time is set to be 2 hours. The showed results are coming out of

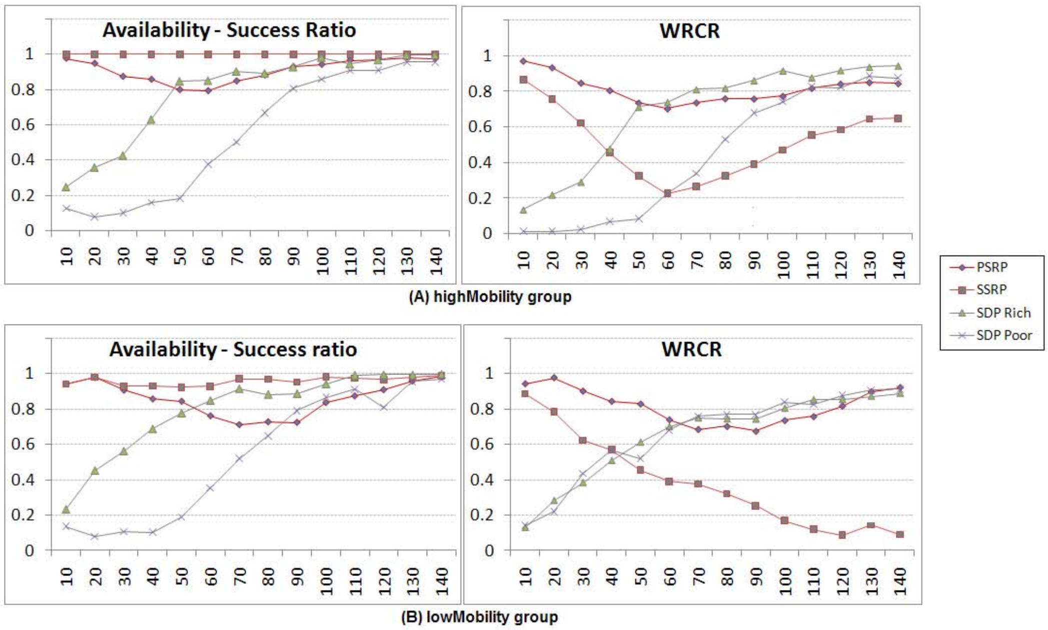


Figure 9.8.: Service availability and WRCR of PSRP, SSRP and SDP Success Ratio (2 client groups) vs. the network size for the two mobility groups

20 runs. The performance matrices are evaluated against different network sizes varying between 10 and 140 nodes.

9.5.2. Results and Analysis

PSRP and SSRP performance analysis:
First, for the "highMobility" group, as shown in Figure 9.8-A, for PSRP, the service availability is high when the network size N is low (i.e., when $N <= 20$). Then, it starts to decrease with an increasing in N (i.e. when $30 < N < 60$), and finally it increases with any further increasing in N (i.e. $N \geq 60$). This can be explained as follows: To predict network partitioning, PSRP is designed to be triggered at a determined time interval, σ, ahead of a possible partitioning. PSRP works efficiently when the partition is affected by one topological change. In case of multiple topological changes (i.e., a new topological change occurs before PSRP has terminated), PSRP restarts execution from scratch whenever a new topological change. If topology changes do not stop, PSRP will not terminate. In this case, network partitioning might never be predicted or it is predicted after the partitioning has already occurred. When N is low, the service availability is approximately 1. This is due to the fact that the network partitions are composed only of a few nodes, and

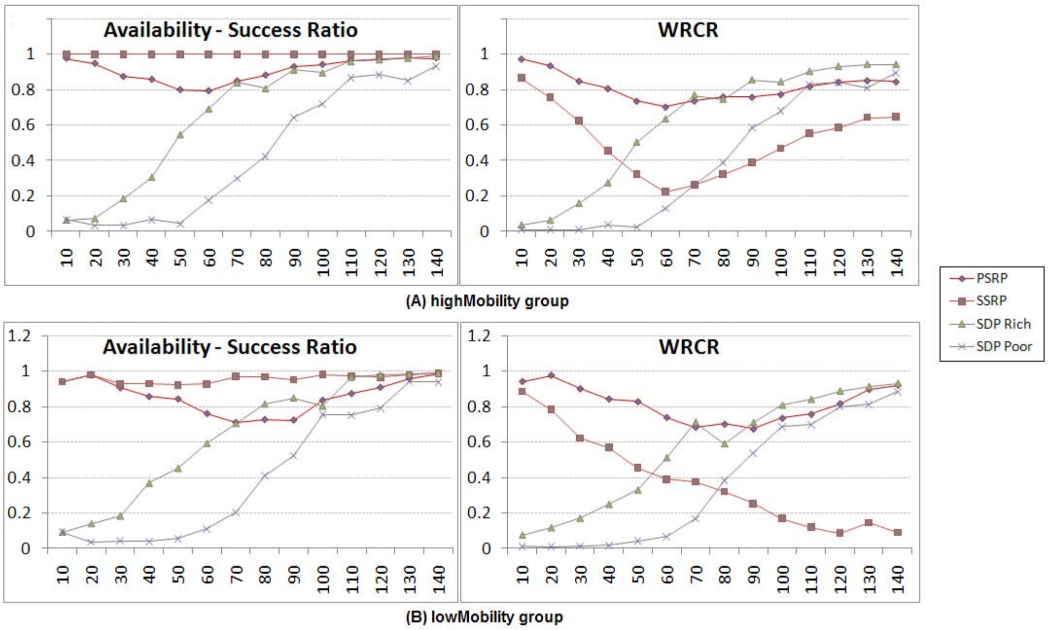


Figure 9.9.: Service availability and WRCR of PSRP, SSRP and SDP Success Ratio (4 client groups) vs. the network size for the two mobility groups

in almost cases PSRP can predict the partitioning within σ time units. When N increases ($30 < N < 60$), multiple concurrent changes occur frequently and hence PSRP becomes less accurate in predicting future network partitioning, which decreases the service availability. After a certain level N: ($N \geq 70$), it becomes very high, and most of the nodes belong to large connected partitions. Although nodes move around, high node density means that partitions remain connected for longer durations, which means less multiple topological changes and high service availability. As SSRP is designed to handle concurrent topological changes and to be accurate in predicting network partitioning, it can ensure the highest service availability. SSRP produces higher numbers of replicas than PSRP (higher service cost) to deal with the frequently concurrent topological changes quickly. When PSRP becomes less accurate in predicting future partitions (especially when $30 < N < 60$), it fails to produce the necessary servers whereas SSRP succeeds in doing so.
About the correctness of replicas' allocation process which is expressed in terms of WRCR, it first decreases with N for both of PSRP and SSRP. Then after some level, at ($N \geq 60$) they start to increase with any further increasing of N. SSRP's feature of higher service cost comes at the expense of the correctness of replicas' allocation process. Ensuring higher availability against the multiple concurrent formed partitions leads to a worse replica allocation process. SSRP's WRCR continues collapsing as higher number of small partitions appear (see the partition analysis for the same network settings in Chapter 7) and more replicas are required. Starting from ($N \geq 60$), the WRCR of SSRP starts increasing because the formed partitions are dense enough to be continued unpartitioned. On the other hand, a smaller number of generated replicas by PSRP ensures that it has a higher corrected replica allocation process with more than 0.81 average WRCR. As a conclusion here, regarding our performance analysis, PSRP can be attributed by a high but not absolute service availability with very high correctness replica allocation correctness ratio WRCR. Ensuring higher service availability by SSRP leads to low correctness replica allocation correctness ratio WRCR.

On the other hand, for the "lowMobility" group, as shown in Figure 9.8-B and in case of PSRP, the service availability is high when the network size is low (i.e., when $N <= 20$). Then it starts to decrease with an increasing in N (i.e. when $30 < N < 70$), and finally it increases with any further increasing in N (i.e. $N \geq 70$). This is can be explained as follows: To predict network partitioning, PSRP is designed to be triggered at a determined time interval, δ, ahead of a possible partitioning. PSRP works efficiently when the partition is affected by one topological change. In case of multiple topological changes (i.e., a new topological change occurs before PSRP has terminated), PSRP restarts execution from scratch whenever a new topological change. If topology changes do not stop, PSRP will not terminate. In this case, network partitioning might never be predicted or it is predicted after the partitioning has already occurred. When N is low, the service availability is approximately 1. This is due to the fact that the network partitions are composed only of a few nodes, and in almost cases PSRP can predict the partitioning within δ time units. When N increases ($30 < N < 60$), multiple concurrent changes occur frequently and hence PSRP becomes less accurate in predicting future network partitioning, which decreases the service availability. After a certain level of N ($N \geq 70$), it becomes very high, and most of the nodes belong to large connected partitions. Although nodes move around, high node density means that partitions remain connected for longer durations, which means less multiple topological changes and high service availability. As SSRP is designed to handle concurrent topological changes and to be accurate in predicting network partitioning, it can ensure higher service availability than PSRP. SSRP produces higher numbers of replicas than PSRP (higher service cost) to deal with the frequently concurrent topological changes. When PSRP becomes less accurate in predicting future partitions (especially when $30 < N < 60$), it fails to produce the necessary servers, whereas SSRP succeeds in doing so.

SDP performance analysis:
First, as shown in Figure 9.8-A, using a maximum requesting rate of 1 request per minute, two client groups will be found which means only half of the network participants will participate in the SDP replication processes at a time. The replication threshold here (1 request per minute) considers that any requesting node is interested in getting a replica from the service. As mentioned before, SDP's performance is expressed in terms of success ratio. For the rich gross interest, starting from moderate number of N: ($N \geq 50$), it achieves a comparable service availability to PSRP. For the poor gross interest in which the service to be replicated does not receive enough interest from the network participants because it is not interesting for most of the mobile nodes, SDP can achieve a better service availability starting from N: ($N \geq 70$). It is very important here to keep in mind that the other service replication protocols except SDP usually consider the offered services are very vital to be replicated (always rich) apart from their importance for the network participants . SDP can estimate a varying importance degree for the offered services and then evaluate its interest-based replication decisions accordingly.
About the correctness of replicas' allocation (WRCR), the rich gross interest achieves the same SSRP's WRCR starting from N: ($N \geq 40$)and PSRP's WRCR starting from N: ($N \geq 50$). For the poor gross interest, although the expected number of the generated replicas is low, the service allocation correctness (WRCR) is generally high and comparable to SSRP's starting from N: ($N \geq 60$).
For the rich gross interest, although the expected (w.r.t. the poor gross interest) number of generated replicas is high, the achieved (WRCR) remained high. This is due to the applied short election mode of the hibernation test, in which the restored

cached services will be quickly (shortly) elected (in terms of interest) to the already active replicas inside its partition.

Second, as shown in Figure 9.9-A, with a maximum requesting rate of 3 requests per minute, four client groups will be found which means that just one quarter of the network participants will participate in the SDP replication processes at a time. The service availability increases as the N increases and decreases as the number of the client groups increases. The availability of SDP in rich gross interest is negatively affected by the four client groups. Again, it is important that we need to keep in mind that by increasing the number of client groups the importance degree of the offered service and the related interest will decrease. In spite of having a less interesting service, it can achieve a high service availability compared to PSRP starting from N: ($N \geq 60$). On the other hand applying a poor gross interest with four client groups simulates a "don't care" attitude of the mobile clients to the offered service. The generated number of requests in this case will be very limited and the number of interested clients will be very low. So, the low offered service availability in this case will be obvious.
In Figure 9.9-A, WRCR is negatively affected by the presented low number of the interested clients regarding the high number of client groups. In case of the rich gross interest, WRCR is generally shifted down of the PRSP's starting from N: size ($N \geq 60$) and is better than SSRP's starting from a moderate N: ($N \geq 50$). For the poor gross interest, WRCR is generally low but starting from N: ($N \geq 70$), it can achieve around and higher WRCR than SSRP's. This means despite of the very low service interest that has been presented in these cases of the higher number of client groups, SDP can maintain a comparable performance to PSRP and SSRP regarding the replica allocation correctness ratio.
The effect of "lower" mobility on all protocols is negatively affecting the service availability, success ratio and the replica allocation correctness. Since both of interest based and topology partitioning prediction based protocols need the service to prevail the network to cover as many as possible partitions, higher mobility behaviors are better to be found.

9.6. Summary

Regarding the structure of this chapter, the conclusions can be drawn in the following two points.

SDP with realistic mobility models SDP has been examined against a set of more realistic circumstances. Two main perspectives have been taken into consideration to provide our investigation namely different mobility models and the service querying behavior. The simulation experiments reflected the hardness of the given network specifications when applying the proposed area graph based driven models especially for the high number of the formed partitions (e.g. very long corridor lengths) and their sizes. SDP showed that it can operate in these realistic situation and preserve its performance compared to SDP results in one place with the simple RWP mobility mode. Moreover, SDP showed a tiny lack of performance when the number of connected corridors changed and stood with different arrangements of settings.

SDP compared to others In the second part of this chapter, SDP is compared as an interest-based service replication protocol for MANETs to two typical partitioning prediction aware replication protocols(namely PSRP and SSRP [DB07, DB08a]). The results highlight the difference between PSRP and SSRP performance and how increasing the service availability can affect the replica allocation correctness. On the other hand, an ascending service availability and replica allocation correctness have been shown by SDP. Regarding the performance analysis, service availability and replica allocation correctness of SDP, PSRP and SSRP can be compared in the intervals of the moderate network sizes. SDP has an advantage of enabling clients to determine their popular services to be replicated. Even for the low popularity services, SDP can deal positively with them. Since SDP does not need to pay any prediction effort for the partitioning behavior of the network, it is slightly affected by the different proposed mobility groups. The effects on SDP performance regarding the different mobility groups were very small and limited compared to the effects on both of PRSP and SSRP. In case of low service popularity (consider the Gross Interest ranges in Chapter 6), SDP can even achieve better replica allocation correctness than SSRP. Regarding the performance analysis and results, SDP can compete with the other network status based service replication approaches which combine their replication decision to the partition prediction and detection schemes.

The previously mentioned two points presented answers for the initial posed questions about the role of mobility and its effects on the service prevalence process in sense of applying some realistic mobility model with SDP. Moreover, the issue about the performance of SDP related to other competitors which are based on analyzing the network status and predicting the partitioning behavior have been addressed and discussed. SDP showed consistent performance in different restrictive environments and compared to other protocols.

Further motivation questions:

Finding ways to further enhance the performance of SDP represents an important motivation. Enhancing the SDP caching mechanism can be a source for enhancements with the service selection process during the service discovery. In order to enhance the replica allocation correctness ratio, a suggestion was to allocate the replicas inactively on the hosting nodes. This idea reduces the number of the active servers in the different network partitions. This can enhance the overall allocation correctness.
The key motivation questions to be considered are:

- How can the nodes without cached replicas benefit from the presence of the cached replicas on other nodes?
- What are the effects of the suggested inactive forwarded replicas on the performance of SDP?

During the next chapter, these questions are considered and investigated.

CHAPTER 10

Enhancing SDP's Service Selection

"If one does not know to which port one is sailing, no wind is favorable"

(Seneca)

The service selection as a process is a main functionaality of the service discovery process in SOAs. Adding new features for the assumed service discovery components (see the general assumptions in Chapter 4 can ensure a better service selection process for SDP. Since SDP toggles the status of replicas (to be active or hibernated) based on their popularities, more dynamicity is required for service selection. In this chapter, a set of different service selection strategies for SDP is introduced. Moreover, the effects of these strategies on the service distribution correctness has been measured, compared and discussed. Finally, in order to show a better illustration for the achieved performance enhancements, we re-compared the SDP with some of these proposed service selection modes to PRSP and SSRP [DB07, DB08a] protocols results of Chapter 9.

Replica allocation correctness is dynamically changing under influence of the ever-changing natures of MANETs. So, the continuous required leader election and service selection are continuously affecting the replica allocation process. In this chapter, we will complement the SDP leader election modes which are given in Chapter 8 with new service selection mechanisms, modes, to ensure higher service availability and better service allocation process. Two new service selection modes have been discussed and investigated through a detailed elaborated simulation. The results have been analyzed and the resources of performance enhancements have been identified. The work in this chapter has been partially published in [HKR10c] and has been done within our cooperation with Dr. Abdelouahid Derhab from Department of Computer Engineering, CERIST Center of Research, Algiers, Algeria.

The structure of this chapter is as follows: In Section 10.1, the new proposed service selection modes for SDP are proposed and discussed. An evaluation based on a detailed simulation experiment is introduced in Section 10.2. Then, the results are presented and analyzed. Finally, the work of this chapter is summarized, the contribution are highlighted, and the next research motivations are presented in Section 10.3.

10.1. SDP Service Selection

As assumed, the service discovery component should provide the required information for clients about the reachable active services in their network partitions. The discovery process is based on the available repositories and the active service offers. Caching services at the hibernated service providers increases the local availability of them for their caching parties. This local availability can be generalized. Enabling the discovery components to store, search and deal with the hibernated service offers can enrich the service selection. In this chapter, we evaluate the effects of enabling the clients to activate some hibernated service provider. Moreover, we

investigate exploiting of this generalized availability in increasing the correctness of the replica allocation process also.
In general, SDP service selection is performed from two perspectives. The first one highlights how a set of clients can decide which service is more interesting and then they direct their requests to it. Upon this decision, the second perspective is included, in which, the replication process can evaluate how and where to activate a replica if there is enough service interest.
In the first perspective, the two long and short SDP leader election modes have been introduced and discussed in Chapter 8. Then, here in the second perspective, two service selection modes of "Wakeup-Neighbors" and "Forward-Inactive-Replicas" are proposed.

The different concepts of SDP proposed service selection modes are introduced in the state diagram in Figures [10.1-10.3] (considering both of Figures 8.1 and 8.2 of Chapter 8). As mentioned in Chapter 8, the states mean:

- A connected client: a client which can achieve a communication to any of the service providers.
- Active service provider: Any mobile host which has an active service or replica and being ready to respond to the client requests.
- Service in evaluation: The time that a service provider check the number of gained requests of a specified service.
- Provider with a hibernated service: A mobile node that caches a service or a replica.
- Generates a neighbor wakeup request: Any unconnected client which detects the presence of a "provider with a hibernated service" and has an interest to query its cached service.

On the other hand, the transition conditions contain a set of important constraints like:

- N/A Service provider: No service provider can be reached by a specified mobile host.
- A/V Service provider: At least one provider can be reached by a specified mobile host.
- A request: A client generates a request for a specified service, dominated by its requesting mode.
- Gained requests: At a provider side, the number of received requests during a certain time interval.

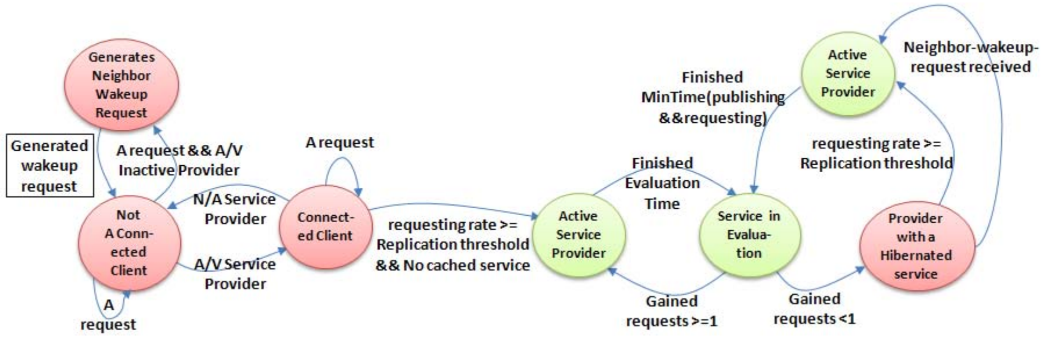


Figure 10.1.: Wakeup-Neighbors service selection mode

- Cached service: The mobile node has a hibernated service.
- Inactive provider: a mobile with a hibernated service.
- MinTime(publishing-requesting): The minimum required time for publishing a new service or activating a hibernated service and getting at least one service request.

10.1.1. Wakeup-Neighbors Service Selection Mode

In some cases, an interested client in a specified service may be found in a network partition where no active services are available although, there may be some hibernated service in this partition. SDP in these cases uses the information available about the hibernated services in the available service repositories. This information enables the interested client to send a "wakeup" request for any of the hibernated service. This request will activate at least one hibernated service. As shown in Figure 10.1, a new state of "Generates Neighbor Wakeup Request" has been added, it is toggled by an unconnected client which detects the availability of an inactive service provider. On the other hand, a new transition which makes an inactive provider respond by activating its service upon the arriving of the wakeup request. Regarding the assumed presence of service repositories, it is not arbitrary and usually are supposed to be found in service-oriented-enabled MANETs. Moreover, it can not be considered as a required infrastructure for SDP because it is possible and feasible that each of the clients may maintain a database that contains the required information form the service discovery module about the offered reachable service (active and inactive).

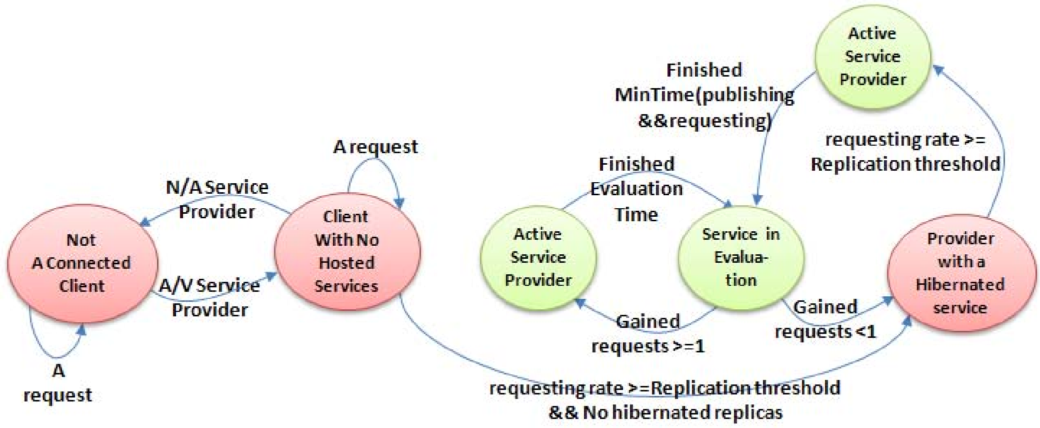


Figure 10.2.: Forward-Inactive-Replicas service selection mode

10.1.2. Forward-Inactive-Replicas Service Selection Mode

In order to increase the correctness of the replica allocation process, the idea of replicating inactive replica (hibernated) at the hosting sites and then it becomes activated if there is enough interest, as in Figure 10.2, is a candidate. Generally, the lower the number of active replicas inside the same partition is, the higher the WRCR ratio is. This idea is used in most the replication protocols like [DJ07, DB07]. The suitability of applying such an idea for an interest based service replication protocol is going to be evaluated in this chapter.

10.1.3. Wakeup-Neighbors and Forward-Inactive-Replicas Service Selection Combined Mode

Finally, as shown in Figure 10.3, a combination of the previously mentioned service selection modes is presented here to magnify WRCR as possible.

10.2. Evaluation

A detailed simulation has been elaborated based on the simulation tools of Appendix A. As mentioned before, a better achieved allocation correctness can enrich generally the performance of any replication approach. In this work, we include in our analysis the following performance metrics.

- General Service Availability (for PSRP and SSRP): as mentioned in Chapter 9.
- Success ratio (for SDP).

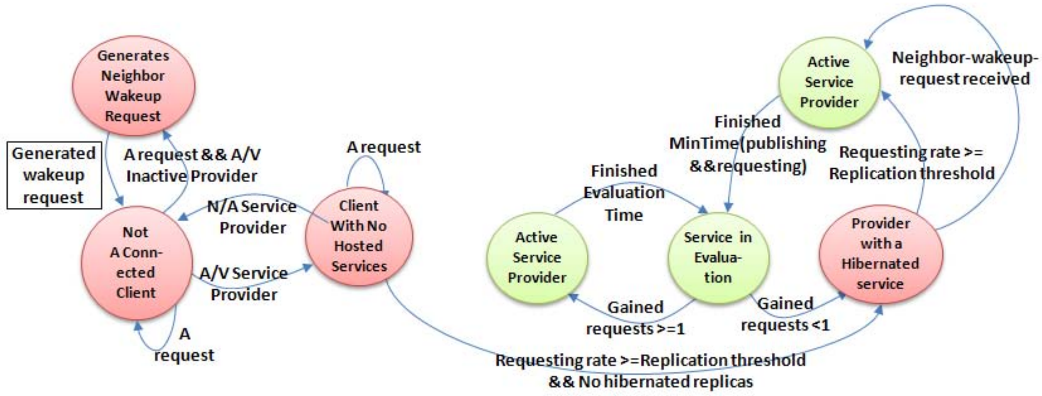


Figure 10.3.: Wakeup-Neighbors and Forward-Inactive-Replicas combined service selection mode

- WRCR.

10.2.1. Settings

The network participants are placed in a $600m \times 600m$ square area with no obstacles where the Random Waypoint mobility model [BHPC04] is applied, in which each node selects randomly a destination in the given area and a speed between 0 and $6m/s$, then it stays fixed during a pause time between 0 and 15 seconds before selecting a new random destination and a speed and so on. Each mobile node can cover a 75 m radio transmission range about itself. The simulation time is set to be 2 hours. The showed results are coming out of 20 times runs. The performance matrices are evaluated against different network sizes varying between 10 and 140 nodes.

A short-mode for the leader election is applied in all of the experiments with one second time interval required for the publishing process. Otherwise the hibernation check interval is one minute. The replication threshold is set to be $|MaximumRequestingRate| : 1minutes$, which means that a node which achieve number of request equals the maximum allowed requesting rate has the right to request its own replica. The Requirements' Index (see Chapter 6) varies its values 20% normally about a general index of 0.5. The client requesting behavior by the proposed Gross Interest scenarios (rich and poor) of Chapterchapter6:grossInterest. Two different client groups (2 and 4 client groups categories) are used. These groups differ in their maximum requesting rate. In cases of 2 Client groups, the maximum requesting rate will be 1 request/minute (client groups: 0 and 1 request/minute). In cases of 4 Client groups, the maximum requesting rate will be 1 request/minute

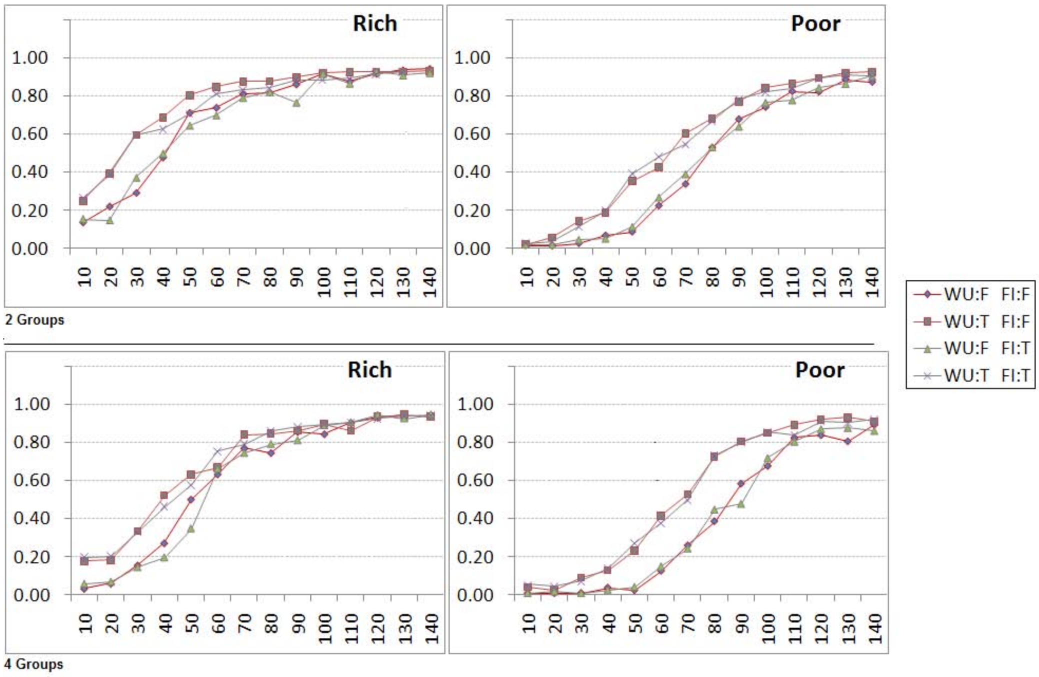


Figure 10.4.: WRCR vs. the network size

(client groups: 0,1, ... 3 requests/minute). The replication threshold is set to equal the maximum requesting rate.

10.2.2. Curves' Naming

In Subsection 10.2.3 of the primary analysis, the curves which are shown in Figure 10.5 and Figure 10.4 are labeled based on the service selection option that they reflect. The new service Selection modes abbreviations are WU for the "wakeup-neighbors" selection mode and FI for the "forward-inactive-replicas" selection mode. For example, a label of "WU:F FI:T" means that the "wakeup-neighbors" modes is deactivated (F: false) and the "forward-inactive-replicas" mode is activated (T:True). So we have four curves { WU:F FI:F, WU:T FI:F, WU:F FI:T, WU:T FI:T } which represents respectively {the proposed service selection modes are inactive, only the "wakeup-neighbors" mode is activated, only the "forward-inactive-replicas" mode is activated, and finally, the combined mode is activated}.

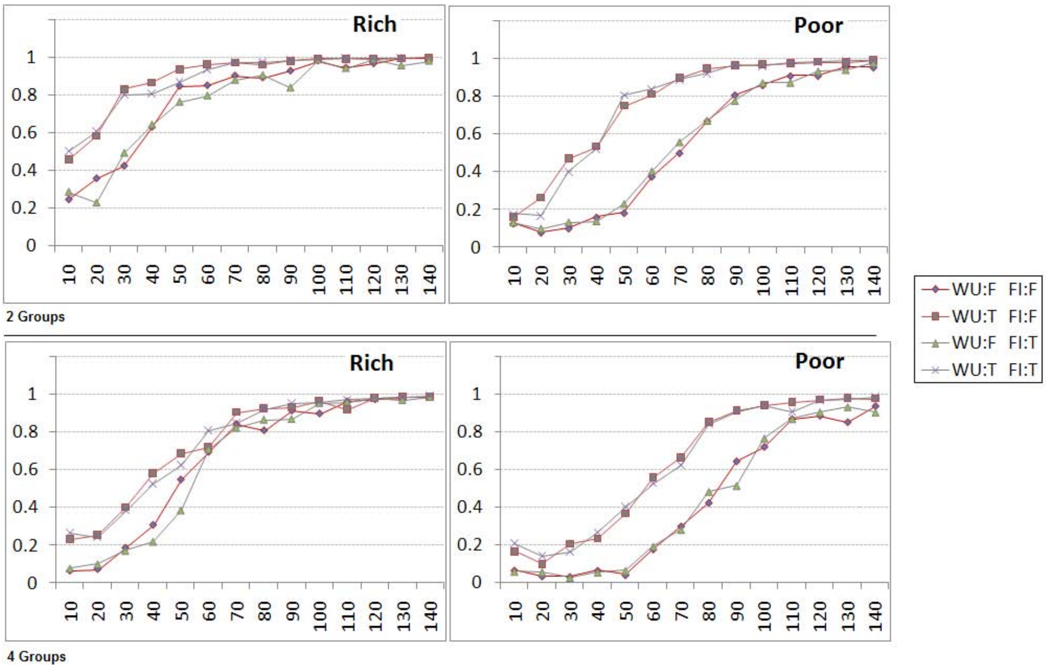


Figure 10.5.: Success ratio vs. the network size

10.2.3. Primary Analysis

In Figure 10.4, we can conclude that applying only the "wakeup-neighbors" mode ensured higher correct WRCR ratio for all Gross Interest scenarios and client groups' number. Even in case of an uninteresting service, the "WU:T FI:F" experiments showed the best enhancements regarding the WRCR ratio. In the rich Gross Interest scenarios, these enhancements are about (8%) of WRCR with average decreased standard deviation of (10%) for the 2 client groups and about (8%) of WRCR with average decreased standard deviation of (2%) for the 4 client groups. In the poor scenarios, the enhancements become more clear. They are about (11.3%) of WRCR with average decreased standard deviation of (3%) for the 2 client groups and about (1.5%) of WRCR with average increased standard deviation of (1%) for the 4 client groups.

On the same behavior patterns as mentioned in the previous figure, in Figure 10.5, applying just the "wakeup-neighbors" mode ensured a better success ratio for both of "WU:T FI:F" and "WU:T FI:T" experiments and both Gross Interest scenarios. Applying only the "forward-inactive-replicas" mode does not ensure any higher success ratio as shown in the "WU:F FI:T" experiments. In Figure 10.5, the highest overall enhancement in the success ratio values are resulting from applying

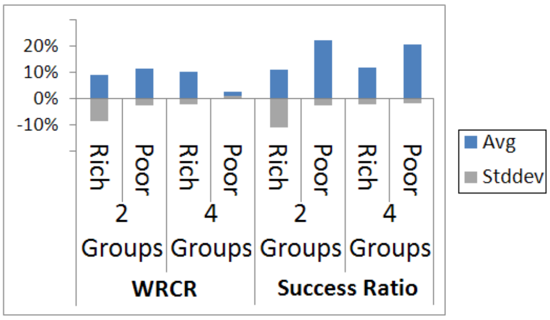


Figure 10.6.: The average enhancement of applying the “wakeup-neighbors” mode

only the “wakeup-neighbors” mode. On the first hand, in the rich scenarios, the overall average enhancement between the curves of “WU:T FI:F” and “WU:F FI:F” is (10.97%) for the 2 groups and (11.41%) for the 4 groups experiments. Moreover, the noticed standard deviation has been decreased in these experiments by (11.14%) and (2.54%). On the second hand, in the poor scenarios, for both 2 and four groups, a significant variance in the effects of the proposed modes is clear. Applying the “wakeup-neighbors” mode can keep highersuccess ratio against presence of low or uninteresting service. The overall average enhancement between the curves of “WU:T FI:F” and “WU:F FI:F” is (22.15%) for the 2 groups and (20.34%) for the 4 groups experiments.

These showed results highlight that the effect of applying the “wakeup-neighbors” mode can enhance WRCR (which indicates the replica allocation process) and the resultant success ratio (which indicates the service availability), especially, in the low Gross Interest scenarios.

As shown in Figure 10.6, the overall achieved enhancements for both the WRCR and success ratio of the “WU:T FI:F” compared to “WU:F FI:F” experiments are presented. It can enhance the WRCR ratio in average about (8%) and the success ratio with an average about (16%). Although the WRCR enhancements are relatively lower than the enhancement in the success ratio, WRCR’s enhancement leaded to the relative great success ratio enhancement.

10.2.4. Extended Analysis

As previously mentioned, applying only the “wakeup-neighbors” service selection mode can enhance the overall service availability and replica allocation process. In this section, we include two of the traditional partitioning-aware protocols which are PSRP and SSRP [DJ07, DB07]. In Figure 10.7, a comparison between the PSRP and SSRP and availability against the SDP success ratio is presented. SSRP can always achieve the best service availability while its WRCR is clearly low. This is

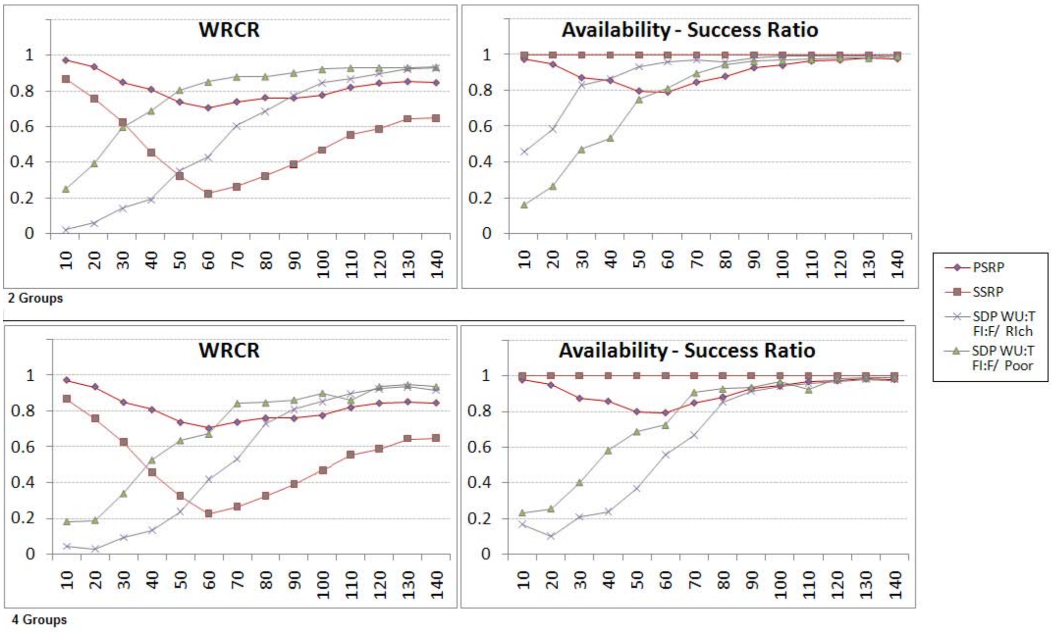


Figure 10.7.: Service availability and success ratio of SDP, PRSP and SSRP vs. the network size

due to its nature of producing and activating higher number of replicas against the varying in the network topology compared to PSRP which achieve better WRCR but on the account of the service availability, nevertheless, PSRP can achieve a high service availability as well. For 2 client groups, regarding the success ratio, both of the applied rich and poor scenarios enabled them to quickly overcome PSRP and then SSRP availability. Starting from low network sizes (30 nodes at a rich scenario and 50 nodes at a poor scenario). Generally, both rich and poor scenarios achieve higher WRCR than PSRP and SSRP. Starting from moderate network sizes of nodes SDP is achieving higher WRCR ratios. On the other hand, for 4 client groups, regarding to success ratio, SDP can achieve a comparable performance compared to the others starting from a moderate network sizes (50 nodes) for both Gross Interest scenarios. For both Gross Interest scenario, SDP can achieve better WRCR ratios over SSRP starting from a low network size (30 nodes) and over PSRP staring from about (50 nodes).

10.3. Summary

In this chapter, two proposed service selection modes ("wakeup-neighbors" and "Forward-inactive-replicas") and a combination mode of these two modes have been introduced. The enhancements in SDP's replica allocation process regarding each of these modes have been shown and discussed. The enhancements in the replica allocation process (in terms of WRCR) lead to great enhancements in the service availability (Success ratio, taking into consideration the service interest) especially, in the poor service interest scenarios which indicate presences of uninteresting service. This represents an advantage of SDP. It means that SDP able to maintain a more correct service distribution even for the uninteresting service (low number of generated service requests). Starting from moderate network sizes (about 50 nodes), it can achieve a equal or higher $WRCR$ ratio than the compared PRSP an SSRP protocols. For the poor scenarios, SDP with the new service selection modes can also maintain a comparable service availability (success ratio), starting from moderate network sizes (about 50 nodes), in spite of evaluating uninteresting service for most of the participants.
As mentioned in Subsection 10.1.2, evaluating the "forward-inactive-replicas" mode showed that it is not a suitable candidate for SDP as an interest-based service replication protocol for MANETs. It has not any positive effect on either the replica allocation process or the service availability. Since in SDP, clients are responsible to evaluate the degree of importance for an offered service, the presence of active replicas enables clients to determine the most interesting offered replicas which will prevail the network in a self-adaptive manner. It has been shown that applying the "wakeup-neighbors" mode (either alone or combined with the "forward-inactive-

replicas" mode) can provide a better service allocation process and higher service availability.

In this chapter, we have evaluated a possibility of enhancing the SDP performance from many aspects. Regarding the achieved enhancements, the "wakeup-neighbors" service selection mode showed promising results. During the next chapters, it is shown how this mode can be utilized in more realistic environments where the resources in the network are not usually available for SDP core mechanisms.

As mentioned in Chapter 4, abundance of resources at the client side is assumed to enable the process of hosting and operating the deployed replica once required. In reality, this assumed abundance is very rare. The idea of the "wakeup-neighbors" is presented as one of the contributions by this chapter to enable a mobile node to activate a cached replica of another node. Actually, considering this idea in situations where there are not enough resources by the servers to serve new requests may help. If a client with a "busy" response can find another server (active or with cached replica), ask it to activate its replica (if it is cached), and pose the requests to it, this may enhance the overall performance in these cases.

Further motivation questions:

The key motivation questions to be considered in cases where there are rare of resources can be addressed as follows:

- What are the levels of resources to be considered? From which perspectives? In which scenarios?
- How can we apply the different SDP selection modes to provide whatever required mechanism in cases of the scenarios of restricted resources?

In the next chapter, the previously mentioned motivation questions are considered and investigated.

CHAPTER 11

Resource Awareness for SDP

"One who chases after two hares
won't catch even one"

(A Japanese proverb)

The assumptions made in Chapter 4 about: (a) the ability of a single service provider to satisfy all the generated requests in it local network partition and (b) the availability of the required resources of a service to be hosted and operated by each of the network participants, are not feasible in the realistic MANETs. Both previously mentioned assumptions are violating directly the concepts of resource awareness. In this chapter, our objective is to provide the mechanisms of SDP of service replication with resource awareness.

The first assumption requires very powerful service provider machines which can offer many resources for the unlimited client requests in its partition. Practically, most of a MANET's participants are relying on a very limited set of resources with limited computation power capabilities. If a real SDP implementation is applied without a consideration for the provider capabilities, it will lead to presence of congestion at the provider sides which will affect directly the quality of the delivered functionalities. In the first part of this chapter, an interest-based load balancing mechanism to distribute the workload over many provider sides is introduced for SDP. The effects on SDP performance and its replica allocation process have been investigated.

The second assumption is the same as the first assumption. It requires unlimited availability for the resources that a replica needs on the provider side to be found in all the network participants. What if a the resources at the network participants vary and some of the participants can not participate in the service replication process? The answer to this important question can provide a better illustration for SDP performance in realistic situations. In the second part of this chapter, the effects of having services competing for resources have been investigated and presented to reflect the expected performance of SDP in less-resourced operation scenarios.

In this chapter, the Opnet® technology has been used in our investigations. Release notes, manual, programmer guide and sample projects of the SDP implementations in Opnet Modeler Wireless® have been published in the Opnet contributed models web page[1] [2] and available for download. More details about these implementations are presented in Appendix B and C .

As previously discussed (in Chapter 4), One replica is assumed be able to satisfy all the requests of the interested clients inside its network partition. Actually, such an assumption can not be always valid. In more real situations, based on the available recourses at the providers, serving new requests is deeply affected by the current utilization of the provider resources and the applied quality schemes.
Assume a scenario of a dense MANET with only one vital (very interesting or popular) deployed service. Suppose this service requires a defined interval of time, a

[1]`https://enterprise1.opnet.com/tsts/4dcgi/MODELS_FullDescription?ModelID=944`
[2]`https://enterprise1.opnet.com/tsts/4dcgi/MODELS_FullDescription?ModelID=951`

certain amount of power, processing, portion of memory, and disk space to meet a single request. Considering the limited resources of the mobile nodes, the service provider at a moment will probably utilize most of its resources to produce the required service responses. If the provider can not accept any new service requests at certain time or can not satisfy the required quality of responses, then SDP's assumption that one replica per partition sufficiency is violated. A direct solution for this challenge is dividing or distributing the requests at the highly loaded provider sides to other providers as possible.
The SDP replica allocation correctness is based in general on previously mentioned assumption of the sufficiency of one replica in a partition (see Chapter 7). If dividing the request load over many sides is considered as a solution in such a case, the replica allocation correctness computation methods from Chapter 7 need to be modified to reflect efficiently the correctness of service distribution in presence of more than one running replicas inside the same network partition. In this chapter, we address the load balancing as a solution for the load problems at the service providers. The work of this chapter is being published in [HKR10b].
The structure of this chapter is as follows: In Section 11.1, the related background for the congestion and load balancing and related work are presented. An interest-based load balancing mechanism for SDP is proposed in Section 11.2. Section 11.3 includes a simulation based evaluation for the proposed mechanism. In Section 11.4, the scenarios of the competing services are presented with the related background. These scenarios are evaluated in Section 11.5. Finally, the work of this chapter is summarized, the contribution are highlighted, and the next research motivations are presented in Section 11.6.

11.1. Background

Generally, when the traffic load (mainly cased by the service requests) exceeds the available capacities of the mobile servers (providers), the problem of congestion appears. The congestion problem is influenced by the dedicated load (in terms of requests) at the limited and low resourced mobile servers. Controlling the congestion problem is an important consideration during the design any MANET protocol. Load balancing mechanisms are usually required to avoid as far as possible the effects of the congestion problem in computer networks.

Load balancing is applied in data transportation, routing and service access. A set of load balancing approaches to enhance the IEEE 802.11 MAC Layer caching capabilities in order to avoid the congestion problem are introduced in [BKAL05]. [MS08, SFBH09, RDKK07] introduce a set of congestion avoidance, load balancing schemes and protocols for providing better routing functionalities in MANETs.

Distributing the workload of accessing some services (like Internet Gateways IGW) in MANETs get increasing research effort. For example, [KAY+07] provides a categorization for the IGW load balancing mechanisms and holds a comparison between them. It also introduces a mechanism based on the network topology to enhance the overall network throughput. Based on the ratio of traffic to the offered bandwidth, [HLT04] proposes a two-tier, heterogeneous MANET architecture which can support Internet access by a set of solutions, namely boundary-moving, host-partitioning, and probabilistic solutions, to solve the load-balance routing issue to the services. [LTEST08] presents an overview of the required functions for providing Internet connectivity and mobility management for mobile ad-hoc networks (MANETs). It introduces a hybrid metric for Internet gateway selection. The hybrid metric provides load-balancing of intra/inter-MANET traffic with a better performance in terms of packet delivery ratio and transmission delay.

[LL04] introduces a survey and taxonomy for the load balancing techniques in the context of grid computing. This taxonomy can be easily generalized to the context of services in MANETs. Three main measures are used for the classification: (a) Static vs. dynamic load balancing: The static balancing is based on some prior knowledge and used statically to trigger the mechanism. In dynamic load balancing, the required information for triggering the balancing mechanism is dynamically collected. (b) Centralized vs. distributed: It depends on a specified centralized manager for controlling the load balancing mechanism. (c) (Service) application-level scheme vs. system-level: while the application-level load balancing focuses on minimizing the required processing time, the system-level load balancing maximizes process throughput or the overall utilization rate of the machines. Regarding this taxonomy, the interest-based load balancing mechanism proposed in this work can be classified as a dynamic distributed service-level load balancing mechanism.
The first part of this chapter, Sections 11.2 and 11.3, highlight our solution for the SDP request congestion problem using a load balancing mechanism.

11.2. The Proposed Load Balancing Mechanism

In this section, we first discuss the dimensions which affect the proposed mechanism in the first two Subsections 11.2.1 and 11.2.2. Then, a detailed description for the proposed mechanism is given in Subsection 11.2.3 .

11.2.1. Connected Sessions to a Service

Each service request initiates a request session once it reaches the provider side. Within an established connected session, the required resources for this request to be processed are allocated and mounted. Moreover, service invocation and execution are done (partially) based on the session features.

In this chapter, the number of established connected sessions to a specified service provider is used as an indication for the resource utilization at this provider. A higher number of connected sessions at a certain provider at a time indicates how busy this provider is.
Based on the resources consumed by a given service regarding a request and the availability of these resources at a hosting provider, a limit for the maximum number of allowed connected sessions can be computed. This maximum limit can be fixed or varied based on the service provider preferences during the network operation time.

11.2.2. SDP Interest-Based Service Selection

The proposed load balancing mechanism is mainly based on the following SDP components:

- A "wakeup-neighbors" mode is introduced in Chapter 10. Based on the presence of a service offer repository and based on an assumed discovery process for the ex-providers with hibernated cached replicas, a client can compose a "wakeup" request for one of those ex-providers in case that there is no active reachable provider in the current partition of the client.

- Service popularity ranking and the Gross Interest:
  SDP assumes (see Chapter 6) the presence of popularity ranking which is not only based on the match values (functional and non-functional) of the matchmaking components between a service request and offers but also on other factors such as a variant importance degree (popularity) of this service to a set of clients at certain time.

- SDP Hibernation Mechanism:
  Since in SDP, all clients are required to direct their request to the most interesting service provider (or a set of providers, as we are going to show), the others will hibernate their replicas. Assuming that the overall request load generated at a time is too huge to be served by one provider, then more than one service provider is supposed to continue and the clients should distribute their request to this set of providers. Of course, this requires modifications to the SDP core mechanism of hibernation to be more suitable for the proposed load balancing mechanism.

So, we can summarize the challenges to be attacked by our proposed mechanism into the following points based on the service interest:

- Allowing a set of providers to continue working apart from the applied hibernation specifications in case of high workloads at the provider side.

- Enforcing the clients to pose their requests to a set of providers based on their popularity ranking rather than posing these requests to the most interesting or popular replicas.
- Measuring the replica allocation correctness of a load balanced service distribution.

11.2.3. Interest-Based Load Balancing Mechanism

As mentioned in Subsection 11.2.1, by using an explicit limit for the number of session, the proposed mechanism can manage the load distribution over many provider. The proposed mechanism gives a role for more than one provider in the environment. The main actors involved are as following:

- Loaded service provider
- Unloaded service provider
- A provider with a hibernated replica
- Interested client

Figure 11.1 shows a scenario where the proposed mechanism is applied. Once an interested client gets a busy response of a loaded provider, it triggers the service discovery to find the next available interesting provider based on service interest ranking inside the same network partition. The next found provider may be an active or hibernated service provider. The upper right part of the figure shows a normal request-response behavior for an active unloaded service provider (Case1). If the found active service provider responded with a busy response, for any reason, the client triggers the service discovery to find the next active provider to this "loaded" provider. Practically, finding more than one active loaded service provider in the same network partition (in spite of the applied hibernation and SDP selection strategy) can be due to the application of the proposed load balancing mechanism. In case of a dense network with very frequent number of requests and very restrictive limit of the maximum allowed number of connected sessions, a client may try several times to search for an unloaded service provider.
If there is no active provider inside the client's network partition, the service discovery component finds the most interesting hibernated service provider. Based on the service selection mode of "Wakeup-Neighbors" which is presented in Chapter 10, the client sends a "wakeup" request to the found hibernated service provider. If this service provider can restore, activate and publish its service, an acknowledge will be sent to the client which will forward its request to this provider.
Since the client requests for a specified service can be constrained to be executed

and responded in a certain time depending on the applied quality scheme, the application that is running on the client side can conclude when to stop searching for an unloaded active or hibernated service provider.
As mentioned before, SDP's main advantages result from the service selection strategies which enable it to shut down all the less interesting replicas. The replica that remains active after this selection is the most interesting for all the clients. After applying the balancing mechanism, the activated replicas' providers (for load balancing) have to mark the activated replicas with a special indicator.

The balancing mechanism distributes the extra requests to other providers is based on the available number of the connected sessions that can be served. Provider by provider is activated until this extra requests inside a given network partition are satisfied. Once the number of requests decreases to the limit that does not require all the running service providers, the hibernation mechanism of SDP will automatically close all the unnecessary service providers. The proposed mechanism can preserve the number of deployed providers always below or equal to the limit of the maximum required number of providers to stand with the current service request load.

A modification of the one replica per network partition assumption (as stated in Chapter 7) is required to be made here: the set of active replica providers ("For load balancing") should replace the assumed "super" replica. Since the optimum service distribution is defined in Chapter 7 based on this assumption in terms of $WRCR$ (Weighted Rational Correctness Ratio), we introduce here a new definition for such the $WRCR$ and call it $WRCR_L$ (Weighted Rational Correctness Ratio for Load Distribution) where:

$$WRCR_L = \sum_{i=0}^{n}(\frac{Pz_i}{NetworkSize} \times RCR_L(P_i)) \tag{11.1}$$

where n is the number of network partitions at time t, Pz_i is the partition size of the i^{th} partition RCR_L, and RCR_L is the rational allocation correctness ratio and defined as follows:

$$RCR_L(P_i) = \begin{cases} 0 & R_{i-L} = 0 \\ 1 & R_{i-L} \in \{1, 2\} \\ \frac{3}{R_{i-L}} \times \frac{Pz_i - R_{i-L}}{Pz_i - 2} & R_{i-L} > 2 \end{cases} \tag{11.2}$$

where R_{i-L} is the number of the replicas inside the i^{th} network partition excluding those replicas which have been activated for load balancing.

$$R_{i-L} = R_i - R_L \tag{11.3}$$

where R_i is the number of the active replicas inside the i^{th} network partition and R_L is the number of "for load balancing" marked replicas.

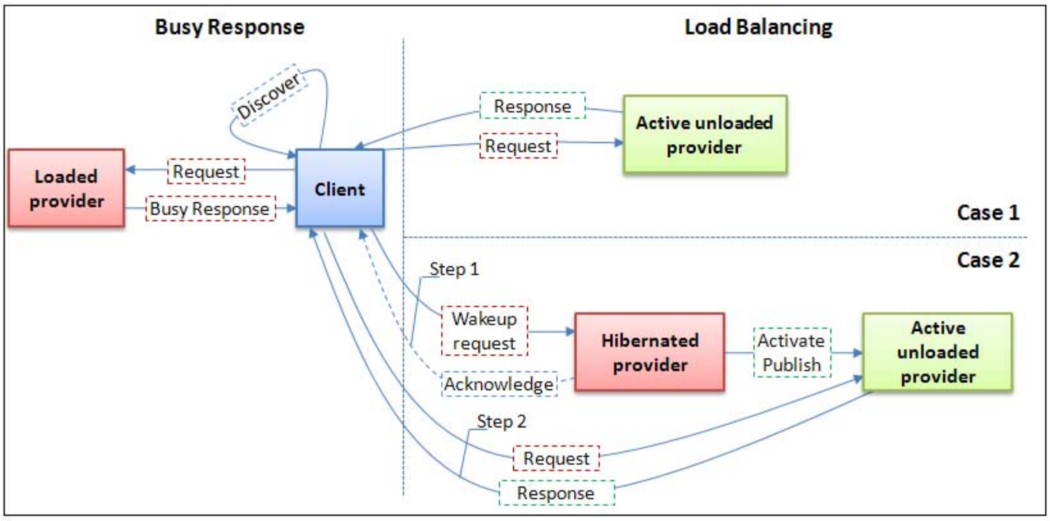


Figure 11.1.: Cases of the interest-based load balancing mechanism of SDP

Listings 11.1 and 11.2 describe in more detail the illustrated interactions in Figure 11.1. In Listing 11.1, an interested client posed a service request to the available service provider. Upon receiving the response ($Response(i, j, k)$ i is the response index, j the client id and k time) and if the response is a busy response, then it tries to find the next (active/hibernated) provider unless the request time is expired.

Listing 11.1: At a client and on a response $Rspns(i, j, k)$ received

```
while (Busy_Response(Rspns(i,j,k)) )
{
  k = CurrentTime;
  j = Find_Next_Provider(Rspns(i,j,k));
  if (j == NULL)
    exit;
  Request = Create_Request(i,j,k);
  Send(Request);
  wait
    Listen();
  until (Arrived_Response(i,j,k)
          ||(Expired_Request(i,j,k)))
}
if (Expired_Request(i,j,k))
  exit;
else
  Process_Response(i,j,k);
```

In Listing 11.2, the request $R(i, j, k)$ reached a provider where i is service id (in case of many services deployed on the same provider) and it is generated by the j^{th} client at time k. This provider may be loaded (can not accept new requests, based on its preferred load limit of maximum number of connected sessions) or unloaded. The function $Accept(< arguments >)$ can determine based on the maximum number of connected sessions, if this $R(i, j, k)$ can be accepted. If the provider is not loaded yet, the request will be accepted, a request response session will be established and the required resources will be allocated.

Listing 11.2: At a provider side (loaded/unloaded) and on a request $R(i, j, k)_t$ arrives

```
if (!Accept(R(i,j,k),t))
{
  Response = Compose_Busy_Response(i,j,k);
  Send(Response,j);
}
else
{
  Session = Create_Session(R(i,j,k));
  Initiate(Session);
  Allocate_Resources(Session);
  Parameters = Serialize_Request_Parameters(Session);
  Invocate_Replica(i,Session,Parameters);
}
```

Listing 11.3: At a provider side and on hibernation check interval expires of an active replica (Rep_i)

```
Gained_Requests = Get_Served_Requests(Rep);
Min_Requests = Nr_Requests(Rep,Hibernation_Threshold);
Run_For_Balancing_Load = Get_Run_Status(Rep);
Check_Hibernation_After =
        Get_Hibernation_Interval(Rep,Hibernation_Threshold);
if (Gained_Requests<Min_Requests)
{
  if((Run_For_Balancing_Load)
    &&(First_Hibernation_test(Rep)))
  {
    Schedule_Hibernation_Check(Rep, Current_Time +
                 Check_Hibernation_After);
    exit;
  }
  Hibernate(Rep);
  Wait_Last_Session(Rep,i);
}
else
  Schedule_Hibernation_Check(Rep, Current_Time
                + Check_Hibernation_After);
```

Listing 11.3 shows how the hibernation mechanism is working together with the load balancing mechanism. If the replica has just been activated this hibernation interval, then it is given a complete hibernation test regardless the applied election mode.

11.3. Evaluation of the Load Balancing Mechanism

A detailed simulation using the Opnet® Modeler® Wireless[3] has been done. The used mobile node model is derived from the delivered Opnet® "mobile_manet_adv" (mobile node) library [OPN]. In the mentioned mobile node architecture, we have replaced the traffic source processor "traf_src" with a new SDP application processor "application". The new added processor does not query any information from the lower network layers of the other processors or components of the mobile node architecture. A project file for SDP with a programmer guide (Appendix B) is available for download at the Opnet® contributed models web page[4] with more details and documentation about the SDP implementation in Opnet®.

11.3.1. Network Settings

The mobile nodes are placed in a $600m \times 600m$ square area. The number of mobile nodes is 50. The Random Waypoint mobility model (RWP) [BHPC04] is applied. The mobile node speeds are uniformly distributed between $[0, 6]$ m/s. The movement pause time is uniformly distributed between [0,15] seconds. An IP addressing scheme (IPv4) is applied with auto assigning mode for the mobile nodes addresses with a UDP data transmission component. The Dynamic Source Routing (DSR) [JM96] protocol for MANETs is used. The used MAC data rate is (11 Mbps) with transmission power of (0.02 Watt). The transmission radius is enforced not to exceed (75 meter). The shown results are collected from the averages of 5 runs (each run simulates a 2-hours-network operation time) with different seeds (seed sequence: Starts from:13213, to: 18000 and step: 1000).

11.3.2. Service Model and Gross Interest Settings

The deployed service to be replicated (distributed) is replicable. It is deployed at the first available mobile node in the network. The required resources to be transfered between the service providers in case of replication is 100KB. The network participants are able to offer the required resources to host and cache a replica.The Requirements' Index 6 varies its values 20% normally about a general index of 0.5 and used in ranking these different replicas.

[3]Version:15.0.A PL1 (Build 8165 32-bit), http://www.opnet.com

[4]https://enterprise1.opnet.com/tsts/4dcgi/MODELS_FullDescription?ModelID=944

In this chapter, the service response time refers to the time interval required for a server to produce a service response. A response time for a service request represents the time interval (life) of its connected session. This time interval is uniformly distributed between [100,200] milliseconds. In case of multiple requests at the provider side, a set of connected sessions will be held. The provider will release the oldest response to be sent to its client in a FIFO style.
As introduced in Chapter 7, since the goal behind the load balancing mechanism is to deal with a high number of requests, the rich Gross Interest scenario is selected to be used in this work. The rich Gross Interest scenario considers the deployed service as a very interesting service. Therefor, it is also assumed to receive a high number of requests by the clients. The short election mode (see Chapter 8) is the basic provider evaluation for the hibernation mechanism with an active "wakeup-neighbors" mode (see Chapter 10) as a service selection mechanism.

In such a scenario, a very active requesting behavior with relatively short periods of pauses is modeled. The mobile nodes have a *requesting-period-length* equal to 10 minutes and the *max-requesting-units* equal to 5 units. This means that the requesting period could be either 10 minutes, 20 minutes, ..., or 50 minutes. On the other hand, each of the mobile nodes is assumed not to request during the pause periods. Pause period settings are as follows; the *pause-length* equals 3 minutes and the *max-pause-units* equals 5 units. So, the pause periods will be 0 minutes, 3 minutes, 6 minutes ..., or 12 minutes. During the requesting periods, the active client performs the service requests influenced by some requesting frequency or rate. The rate is to be at maximum 1 request per minute.

11.3.3. Performance Metrics

In this work the performance of the proposed mechanism is measured in terms of the following metrics:

- General Service Availability: as introduced in Chapter 9.
- SDP Service Availability: as introduced in Chapter 9.
- Success ratio
- Prevalence
- Service Allocation correctness ratio : as previously discussed, in terms of $WRCR$ and $WRCR_L$.
- Session limit: the maximum number of allowed sessions to be connected to a single service provider.

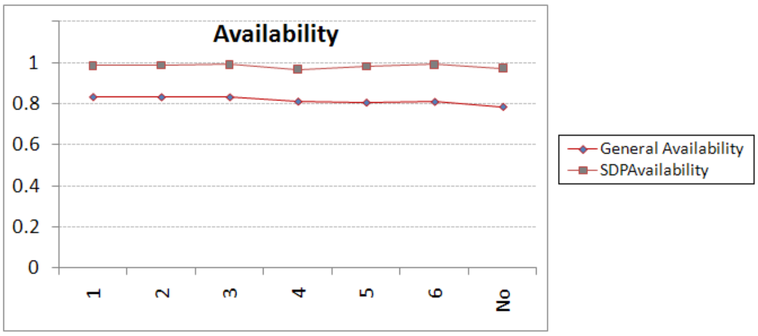


Figure 11.2.: General and SDP availability vs. different maximum number of the allowed connected sessions (session limit, "No" = no limit)

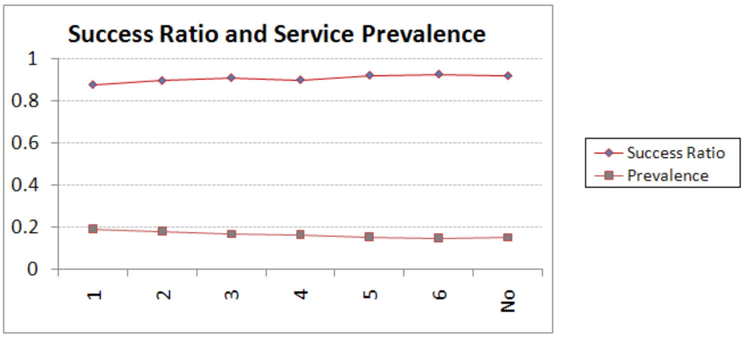


Figure 11.3.: Service prevalence and success ratio vs. different maximum number of the allowed connected sessions (session limit, "No" = no limit)

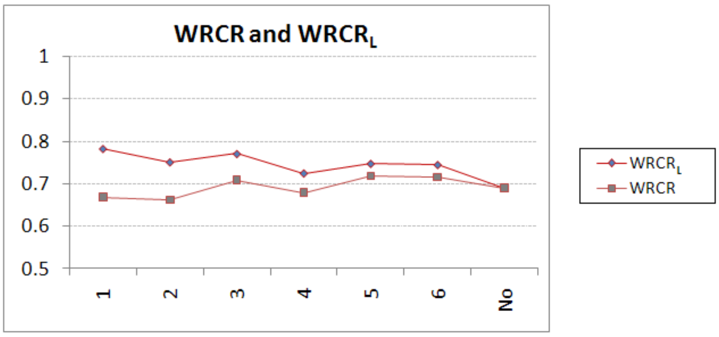


Figure 11.4.: $WRCR$ and $WRCR_L$ vs. different maximum number of the allowed connected sessions (session limit, "No" = no limit)

11.3.4. Performance Analysis

Figure 11.2 shows the effects of varying the session limit on the general and SDP service availability and the resulting success ratio. Regarding the general service availability, as the session limit becomes shorter, it increases slightly. This can be explained by considering the higher number of the remained active replicas when applying more restrictive session limit values. The extreme observed decrease between no and the most restrictive limit (1 session) is about 0.02 with the same value as a standard deviation. So, as an important result to be drawn here, applying the proposed load balancing mechanism has no negative impact on the general service availability and can enhance it slightly.
The importance of introducing the SDP availability can be understood from the needs to indicate the time intervals that the service or any of its replicas was active in the network by the request behavior of the clients. Applying the rich Gross Interest as the requesting behavior of the mobile nodes ensures that the service (the original service or any of the replicas) will be active mostly of the network operation time (see results of service availability in Chapter 7). The results showed that there is also no negative effects on the SDP availability.

In Figure 11.3, the service prevalence has an apparent negative impact of applying the proposed load balancing mechanism and as the session limit becomes more restrictive. Actually, this is due to the definition of the service prevalence as an indication for the service cost [DB09]. The prevalence can indicate the number of the network participants hosting the service or one of it replicas and accordingly the consumed network resources by this service. In case of applying the proposed load balancing mechanism, although the number of hosting nodes increases as mentioned before, the increasing consumption of resources by the higher number of providers is an output of activating the minimum required number of providers to stand with the request load of the clients as mentioned in Subsection 11.2.1. So, the variance of the observed service prevalence should not be considered as a negative impact. The prevalence increase is very limited and in the interval between no session limits and the most tight session limit (1 session) is about 0.03 with 0.005 value of standard deviation. The applied session limit can be considered as the source of negative impacts on the success ratio in Figure 11.3. As the session limit becomes tighter the success ratio decreases. Since the loaded service providers do not accept some of the client requests and accordingly these clients try to find other reachable service providers, some of these requests may be dropped either because of some quality measurement or by the network (e.g. during transportation). The good news is that the decrease in the success ratio is very limited and in the interval between no session limits and the most tight session limit (1 session) is about 0.03 with 0.02 value of standard deviation.

Figure 11.4 shows the advantages of the proposed load balancing mechanisms in terms of service prevalence and replica allocation correctness ratio. The difference between the observed values of both $WRCR$ and $WRCR_L$ can indicate the amount of excess of requests over the maximum allowed limits at a time. As the session limit becomes more restrictive, $WRCR$ decreases While $WRCR$ decreases, $WRCR_L$ increases. Since, the excess of the requests returned from the loaded service providers is redirected to other (active or waked up) providers, the number of the un-hibernated replicas increases. It is very important to note that, in these cases, the number of the concurrently running replicas are necessary to serve the current request load and so, no penalties should be taken against these replicas and that is exactly what $WRCR_L$ does.

In the previous part of this chapter, the importance of having special consideration for the problem of request congestion at the provider sides (where the SDP protocol is applied) is presented. An interest-based load balancing mechanism is proposed. The introduction of this proposed mechanism enables SDP to stand with more realistic restrictive situations. The introduced load balancing mechanism inherits SDP's advantages, since it requires only information about the service interest which can be supplied from the service layer with no need to investigate the lower network layers. The new proposed load balancing mechanism distributes the excess of requests (over the maximum allowed number of connected sessions) to the available active/hibernated providers. Therefore, it allows for a higher number of service providers with similar replicas to run concurrently. For that, a new replica allocation correctness computation method has been introduced. Using the already found service selection methods of SDP (see Chapter 10), the proposed load balancing mechanism preserves a flexible way to deallocate the less interesting service providers if there is no need to keep them active. The proposed mechanism has been investigated through a detailed comparison and the results showed how it can increase the service (General and SDP) availability and enhance the correctness of the replica allocation process. Although the service prevalence increases as the number of the maximum allowed session to the service provider decreases, this should not be a weak point because the higher number of the deployed replicas in that case are required against the presence of the request excess which can be served at the offered service providers' sides.
Our second motivation in this chapter was to investigate SDP's performance in more resource restrictive situations. The important question to be answered is what the SDP performance will be if some of the mobile nodes can not participate in the replication process. To find answers to such a question, the previously mentioned load balancing mechanism can be employed.

In the next part of this chapter, in Sections (11.4-11.5), it is going to be investigated how SDP in a dynamic resource scenarios operates.

11.4. Resource Competing Services

Assume a scenario where two services are deployed in the network. These two services provide different functionalities. A provider can only activate one of these services at any given time. The important question to be investigated here is how can the presence of the resource competing services affect their SDP performance. Moreover, the investigations go further to discover what should happen, if there is a consideration for the request congestion problem and the impacts of applying the proposed interest-based load balancing mechanism.

In the next evaluations, two scenarios of deploying services in the network are considered. The services are deployed in a MANET with same specifications of Subsection 11.3.1. The first scenario considers deploying only one service. So, no effects of competition can be observed. In the second scenario, two resource competing services are deployed in the network. The mobile node included in the network specifications at Subsection 11.3.1 offers 100 KB of memory and 100 KB of disk space. Each of the two services requires 100 KB of memory and 50 KB of disk space. So, they can be cached together, but only one of them can be activated. In case that the interest-based load balancing mechanism is applied, the service providers use the specifications of connected sessions introduced previously.

11.5. SDP Performance with Competing Services Evaluation

A detailed simulation has been made here using the Opnet® Modeler® Wireless and based on the same simulation settings in Subsection 11.3. Also, the used services' model and the Gross Interest settings are the same as in Subsection 11.3.2.

11.5.1. Performance Metrics

The performance during the current investigations is expressed in the following performance metrics:

- SDP Service Availability.
- Success ratio.
- Prevalence.

- Service Allocation Correctness ratio : In terms of $WRCR$ and $WRCR_L$.
- The Typical Partition (Typ.Partition): as discussed in Chapter 7.
- Session limit.

11.5.2. Primary Performance Analysis

As required and mentioned before, the experiments have to answer two main questions: (a) what is the performance of SDP in presence of competing services as defined and (b) what are the effects if not only the resources are restrictive but also the service provider abilities? In order to find answers for these two questions, the results of having two competing services are relatively compared to the results the scenario with no competition "NoCompetition" where only one service is deployed. Figure 11.5 shows the results in terms of SDP availability, success ratio, prevalence and WRCR for different network sizes (10 nodes to 70 nodes). An important behavior in the low dense network sizes (< 40 nodes, except the network size of 10 nodes) is observed. The two competing Services (S0, S1) are mutually toggling their performance measurements. The higher observed performance metrics are for one service, the lower observed performance metrics are for the other service. The partition analysis in Figure 11.6 can provide an explanation. At the very low-dense network size of 10 nodes, the network consists of about 8 network partitions. Deploying the two competing services in such a situation makes most likely each of them hosted in different network partition during the network operation time. Starting form 20 nodes, although the number of partitions increases, the Typical Partition indicates denser formed partitions. So, the probability of having the two competing services deployed in the same partition is higher. Since it is mostly services hosted in the same network partition and considering that the replication process for one of these services will be done before the other one, the service which replicated first a replica will increase its possibilities to prevail in the partition and then the network more than the other service can. Once one of the services occupies more sites, the second possibilities of prevailing the network will directly decrease. This behavior is also noticed at the network densities of 30 and 40 nodes. This behavior affects basically the SDP availability and accordingly the success ratio, prevalence and $WRCR$.
Starting from a moderate network density (at network size = 50 nodes), The Typical Partition indicates denser formed network partitions (about 20 nodes in the Typical Partition). The formed dense network partitions ensure that there are enough mobile nodes that can satisfy the required prevalence for each of the competing services to achieve better performance metrics.
In Figure 11.5, regarding SDP-availability and starting from 50 nodes, the effects of having two competing services on the SDP availability disappear and both services

can achieve about (99.5%) SDP availability with average standard deviation less than about (3%).
Regarding the success ratio, the same behavior of achieving better success ratio as the network becomes denser is observed in Figure 11.5. Starting form moderate network size of 50 nodes the achieved success ratio for both of the competing services are high and above (86%) with a standard deviation of (2%). Comparing the achieved success ratio in the same network size interval of the competing services to the relaxed success ratio (without competition: 94% with 1%) shows the main negative effect of having two competing services on the success ratio in our scenario. In spite of having these negative effects, SDP can achieve in general high success ratios.
Regarding the prevalence and $WRCR$, Figure 11.5, the toggling behavior of the higher prevalence values of the two competing services in the low dense network situations is observed. The lower prevalence observed values come on the account of the correctness of replica distribution process (in terms of $WRCR$) and vice versa. Starting from 50 nodes of networks sizes, SDP starts achieving lower (better) prevalence ratios about average (8-10%) with a standard deviation (4%) with higher $WRCR$ ratios about average (74-75%) with a astandard deviation (20%). Comparing these results to results of the relaxed prevalence and $WRCR$ ratios about averages (Prevalence: 11% and $WRCR$: 76%) shows that SDP remained with high ability to save the network resources (in terms of prevalence) and distribution correctness (in terms of $WRCR$).

11.5.3. Extended Performance Analysis

In this subsection, the questions about applying the interest-based load balancing mechanism with the competing services and its effects are addressed. The proposed load balancing mechanism with its specifications in Subsections 11.2 and 11.3 are applied. The network size is fixed at 50 nodes. The maximum allowed session limit varies between no limit, 1 and 6 sessions.

In Figure 11.7, competition on the available resources at the lower dense network situations is reflected on the SDP-availability. The tighter maximum allowed number of sessions (session limits) leads to produce and activate a higher number of replicas inside the network. Moreover, these activated replicas are required to remain active. Since one of these two competing services will be replicated first, the probability for the second service to find nodes with enough resources is decreased. As the session limit becomes less restrictive, the resources which are required for operating the two competing service become easily available. The shown toggling behavior is not deeply affecting the SDP-availability which is usually above (80%). Starting from session limit of 4 sessions, there are enough resources for the two com-

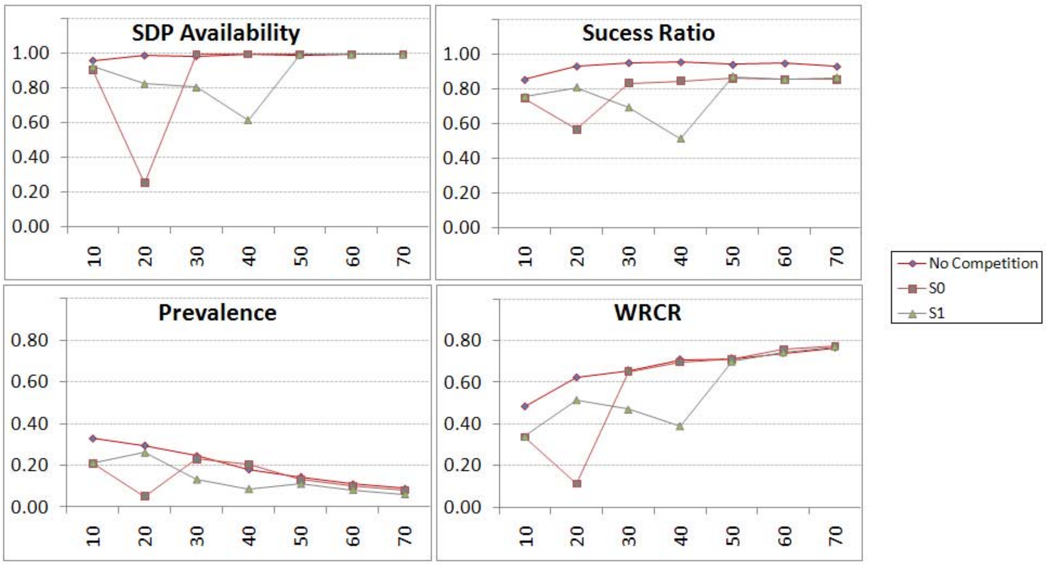


Figure 11.5.: SDP availability, success ratio, prevalence and $WRCR$ vs. varying network size of two competing services S0, S1

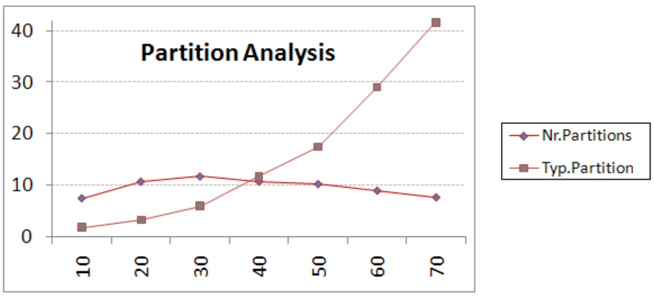


Figure 11.6.: Network partition analysis (network size (X-axis), (the number of nodes in Typical Partition -Typ.Partition- or the number of partitions -Nr.Partitions- (Y-axis)))

peting services to operate with no negative effects of the resources restrictions of the scenarios or the request congestion. Regarding the success ratio, as mentioned previously in Subsection 11.3.4, the tighter session limits have a negative effect on the success ratio as well as the shown effects of the presence of competition for resources between services. The resultant success ratio in low dense network situations of applying the load balancing mechanism with the competing services show a moderate achieved success ratio with an average about (71-72%) with standard deviation (5%). Starting from 4 session limit, SDP can keep a similar performance (in terms of success ratio: about average (82-83%) with a standard deviation about (4%)) for the competing services compared to the "NoCompetition" curve. The achieved success ratio of both competing services remain relatively high values even if the load balancing mechanism is applied.
The relaxed and two competing services prevalence are still very low. In general, the prevalence is less than about (15%) with standard deviation about (5%) for whatever session limits.
In Figure 11.7, the achieved $WRCR_L$ values for both of the competing services follow the toggling behavior of the SDP-availability. Considering that the more tight applied session limit, the better achieved $WRCR_L$ (as mentioned in Figure 11.4), $WRCR_L$ is generally decreasing. Starting from 4 session limit, the average achieved $WRCR_L$ for both of the competing services is about (74-71%) with standard deviation (18%). The overall average of $WRCR_L$ for each service (S0,S1) is (70% and 72%) respectively. These $WRCR_L$s are relatively in the same order of the relaxed $WRCR_L$ which equals (74%).

In the second part of this chapter, the performance of SDP with a resources' restrictive scenario has been investigated. Two levels of restrictions have been applied. In the first level, the mobile nodes' resources to be used during the service replication and operation are not sufficient to host actively both of the services simultaneously at the same mobile node. So, the services (replicas of two distinct services) are competing to prevail on the network participants. In the second level, besides the presence of the first level of restrictions, the mobile nodes and providers are required to apply the proposed load balancing mechanism which has been addressed in the first part of this chapter.
The effects of both levels have been investigated (on two scales: the network size -density- and the session limit respectively). The results showed a small and determined negative effects on both service success ratio, especially, in the lower network densities and the very restrictive session limits. The effects on $WRCR$ and $WRCR_L$ are described and showed how they are (relatively) still high values against the restrictive specifications.

Figure 11.7.: SDP availability, success ratio and prevalence vs. different maximum numbers of the allowed connected sessions (session limit, "No"= no limit) of Competing two services S0, S1

11.6. Summary

In this chapter, the assumptions of the unlimited resource which have been made in Chapter 4 have been deprecated. These assumptions influence two categories of resources in the network. The first category of required resources is the unlimited processing power by the service providers. The second category presents the available resources at each of the mobile nodes to participate in the replication processes. Each of these categories is practically hard to realize in the real network situations.

The problem of request congestion, which violates the assumptions of the first category, has been presented. A proposed interest-based load balancing mechanism that can manage distributing requests over different providers has been introduced. The mechanism's performance has been analyzed. The results showed that applying the proposed mechanism is promising and can increase the correctness of the replica allocation process and contain the excess of the clients' requests dynamically.

For analyzing the performance of SDP in the case that there are not enough resources on the network participants to be replicated in a relaxing way, a network scenario where there are more than one deployed service which competes for occupying the nodes' available resources has been presented. The negative effects of deploying two competing services have been analyzed. The results showed how

SDP can stand with the very restrictive-resources' operation scenarios.

Achieving a higher availability for a set of services together is considered as a challenge for SDP in Chapter 1. Composite service execution represents an important feature for most of the SOA based applications. The following chapter introduces the issues related to composite service execution in MANETs and addresses investigations for these motivation questions. Moreover, SDP performance regarding different execution scenarios is introduced, analyzed and discussed.

Further motivation questions:

The key motivation questions to be considered in cases where composite service requests are considered can be as follows:

- What are the performance affecting parameters on composite service execution based on SDP?
- What are the features of SDP performance with composite service requests?

CHAPTER 12

SDP and Composite Service Execution

"I have no special talents. I am only passionately curious"

(Albert Einstein)

Some tasks require complex functionalities that can not be supplied by one service provider alone. In order to meet these tasks' requirements, a set of distinct (atomic) services which may be deployed on many network participants should be combined. The composite service is a service that contains multiple simpler atomic services. Service composition is the process of specifying and collecting a set of atomic service descriptions in a logical control sequence of executing steps to satisfy a service request. The difficulty of composite service execution in MANETs is the limited availability of the atomic services. Composite service execution forms a higher level of service availability requirements. It extends the meaning of availability to cover a union of services. So, employing an approach that can increase the service availability is vital and tightly coupled to composite service execution. Based on applying SDP as a service replication approach for the atomic services, the required availability can be enhanced and better composite service execution can be achieved. In this chapter, we aim to investigate the limits of performance metrics for the SDP based composite service execution processes in MANETs. We extend our investigations to highlight an SDP based composite service execution approach compared to other approaches. The structure of this chapter is as follows: in Section 12.1 the issues behind the composite service execution in MANETs are introduced and the model of the differently running atomic services is described and discussed. In Section 12.2, a SDP based proposed composite service execution approach is introduced with detailed definitions and assumptions. Section 12.3 evaluates the proposed SDP based composite service approach using a detailed simulation. In Section 12.4, a performance comparison between the proposed approach and other approach is presented. Finally, the work of this chapter is summarized and the contributions are highlighted in Section 12.5.

12.1. Composite Service Execution in MANETs

Recently, service composition gets more research interest. Factors like mobility, device heterogeneity, resource variability and availability of MANETs' features have not been considered in realizing SOA applications in wired-infrastructure based service composition. Therefore, service composition approaches for the traditional environment, like the Web services [New02], can not be directly applied in MANETs. Many approaches have been introduced to solve the problems of service composition in MANETs like [CJFY05, BHI10, PWSK07, JXS07]. Regarding the nature of MANETs, the resultant composite services need to be executed taking into consideration the dramatically ever-changing network status. Composite service execution in MANETs is an important task which requires dynamicity. Let us imagine a BPEL [Jur06] (Business Process Execution Language)-like plan that describes efficiently the inputs/output, resources, interactions and flows between the different services in a composite service. In a MANET, just having such a plan does not imply achieving the required complex functionality, since the offered services are not guaranteed and frequently become unavailable for some or all of the network participants. In this case, flexible execution plans and methodologies are required to realize the composite services. In this chapter, we address and define the composite service execution process with its important features in MANETs. Afterwards, we introduce a composite service execution approach that is based on SDP. A detailed set of evaluations for the proposed approach are presented and discussed.

12.1.1. Service Model

Two types of services are addressed by our approach:
(a) Atomic services: the set of services that can be completely hosted at one provider side who can individually deliver its functionality upon request.
(b) Composite services: a set of services with functionalities that can not be supplied by one service provider only. So, a union of atomic services may compose the required functionality of the composite service. From a client perspective, to get such required complex functionality, a composite service request should be composed based on the available information about the service discovery and choreography. As previously assumed, we do not consider the problem of service discovery and assume that the required choreography information is available when required. A set of atomic services $S = \{S_0, S_1, \ldots, S_k, \ldots, S_x\}$ is deployed in the beginning of the network operation time t_0. Each of these services is deployed on an individual mobile node. Each of these services provides a distinct functionality.

12.1.2. Finding the Execution Path

Since the network topology is ever changing, the paths between the atomic services and clients are varying. From a decentralized perspective of an execution process, the composite service requires a specified flow for the inputs, intermediate inputs/outputs and the final results. This flow is mapped into the wireless links between the network participants. The mapped flow of a composite service is called the *execution path*. The execution path for the process of service execution of any composite service is very important because of its significant influence on the network overhead and the different failure rates. As the execution path becomes shorter, better execution metrics can be achieved [CWSH08]. So, most of the service execution approaches in MANETs aim to minimize the length of execution path for each service composition based on many criteria.
In [CWSH08], the execution path selection approach is based on the type of composite service. If the composite service has a linear pattern (all of the atomic services are called sequentially), the selection can be achieved by using the Dijkstra

algorithm to find the nearest next provider. In case of having some atomic service called in parallel (split and merge pattern), the Dijkstra algorithm is deprecated. The shortest path between each two sequences of services is computed, then the shortest path to an atomic service is computed.
In [SZC+08], the composite services are mapped onto Service Composition Graphs (SCG). The requester node sends for the first atomic service in the SCG. Each node which provides the first service will forward the SCG to any reachable node with the next atomic service and so on until satisfying the atomic services requests in the SCG. A redundancy control mechanism controls the SCGs flow among the different nodes. Finally, a set of SCGs will be sent to the original requester with different proposed execution paths. The shorter paths are selected.
In [CJWS08], a heuristic dynamic execution path selection approach is introduced. The path selection is done dynamically during the execution process. In a decentralized execution manner, the whole composite service request is forwarded to the nearest (in terms of number of hops) first atomic service provider. Then, after executing the first request in the composite service, the current provider finds the nearest second atomic service provider and forwards the intermediate results and the composite service to it and so on until all of the atomic service requests are satisfied. Finally, the results are sent to the original provider.
For the previously mentioned approaches, knowledge is required about the current network topology and status. This knowledge is supplied of the lower network layers and components such as the routing component. As an alternative, the service popularity can propose another method with a centralized execution process. As we are going to show, under the assumption of reflecting how the service is centralized (for example, better response time services are likely to be interesting for the clients and then gain higher number of requests and remain active) for all of its interested clients, the popularity ranking of the different services can be computed. Moreover, in Section 12.4, a performance comparison between the approach of [CJWS08] and an SDP based composite service execution approach is presented.

12.1.3. Controlling the Execution Process

As shown in Figure 12.1, there are two categories of controlling the composite service execution in MANETs: In the first category "**Decentralized**" execution, as in [CJFY05, BHI10], the requesting node ($Node_x$) pushes the initial composite request to the first of the involved intermediate atomic service providers. This intermediate provider processes its related atomic request. Based on some execution plan, forwards the intermediate results, inputs and rest of the composite request to the next intermediate providers and so on. The last involved provider forwards the final results to the requester node after performing the required request process. Since the assumption of the existence of the controller does not fit in MANETs due to their

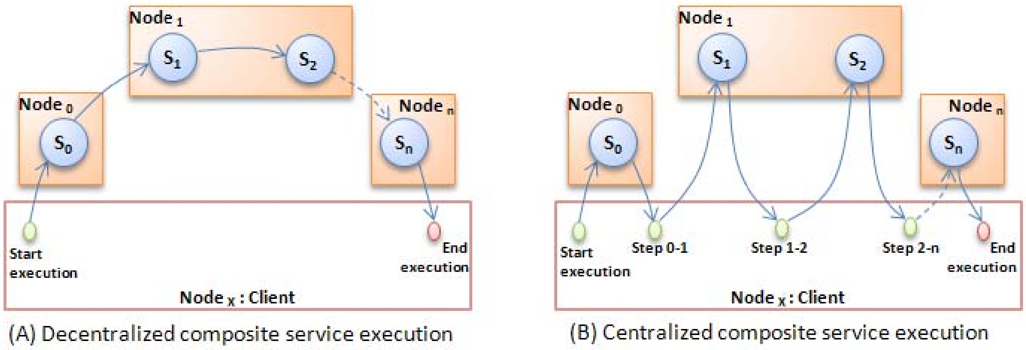


Figure 12.1.: Types of managing the execution plans

decentralized nature and high possibility of single node failure, the decentralized execution is more suitable in MANETs [CWSH08]. On the other hand, each of the atomic service providers in the decentralized service execution should have more resources for caching and forwarding both the composite request and their related intermediate inputs and results. Moreover, the intermediate atomic service provider should work in a proper order on the level of each composite request. Of course, this requires many mounted resources and computations. So, in cases of decentralized execution category, the number, length and frequency of the composite request generation should be taken into consideration. In the second, "**Centralized**" execution, category as in [SZC+08, CJWS08, CWSH08], an assumed party in the network keeps track of the process of controlling the composite service execution. The execution processes here are divided into many execution steps. At each of these steps, an atomic service is requested. The outputs of the previous request is used as inputs in the next execution step and so on. As previously mentioned, it is hard to keep a centralized party that is responsible for controlling the composite service execution. However, taking into consideration the assumption of presence of distributed service oriented architectures like in [dia] and under the assumption of the service oriented application in Section 4.5, we assume (and we think it is a very realistic assumption) that each of the mobile nodes (clients) and at each composite service request generation can play the role of the required centralized party that controls the execution process for its composite requests. By such an assumption, the required resources at the atomic service providers for caching the several inputs, execution plans, and outputs during the decentralized execution can be avoided. Moreover, each of the requesters can control the progress of the composite service execution and flexibly adapt the execution plan at any time while the execution is progressing. The centralized execution saves also the communication links between the network participants by avoiding sending the intermediate inputs, outputs and results several times between the mobile nodes. The required overhead traffic for the controlling process is kept lower. Moreover, if one of the intermediate provider nodes fails or disjoins the network, in cases of the decentralized execution, the whole execution results may be lost. In the proposed centralized execution approach, the client will be somehow informed about the missing intermediate provider node and can redirect the atomic request to another provider node.

12.2. An SDP based Composite Service Execution Approach

12.2.1. Composite Service Execution Definition

Some of the atomic services S_k will have a set of replicas $R_{S_k} = \{R^1_{S_k}, R^2_{S_k}, \ldots, R^i_{S_k}, \ldots, R^y_{S_k}\}$ during the replication process of SDP. The composite service request can be formed as a set CR, $CR = \{S_t, S_o, \ldots, S_v\}$. Based on the required composite request flow, we assume that some information about which requests are required to be executed in sequential order and which are required to be executed in parallel order is included. So, CR can be reformed as a set of sequences $CR_{Execution}$:

$$\begin{aligned} CR_{Execution} &= \{[S_a]_{parallel}, [S_b, \ldots, S_c]_{sequential}, \ldots, [S_d, \ldots, S_e]_{parallel}, \\ &\ldots, [[S_f, S_g]_{sequential}, S_h]_{parallel}\} \end{aligned} \tag{12.1}$$

Consider the network model in Chapter 5. At time t_x, a specified mobile node (client) initiates a service composition process which produces both CR and $CR_{Execution}$. The composite service execution of these sets depends on the current status and availability of the atomic services and their replicas inside the client network partition. By employing a service replication protocol such as SDP, this protocol manages automatically the service selection process based on the previously mentioned popularity evaluation processes. So, translating the given composite request set $CR_{Execution}$ into invocation requests to the individual atomic service providers is considered as interest-based execution plan P. In such an execution plan, the services defined in $CR_{Execution}$ are replaced by the equivalent available replica R^i inside the client network partition. Therefore, P can be defined as following:

$$\begin{aligned} P &= \{[R^i_a]_{parallel}, [R^i_b, \ldots, R^i_c]_{sequential}, \ldots, [R^i_d, \\ &\ldots, R^i_e]_{parallel}, \ldots, [[R^i_f, S_g]_{sequential}, R^i_h]_{parallel}\} \end{aligned} \tag{12.2}$$

Based on Equation 12.2, two performance extremes for the composite service execution plans can be given from the perspective of the order of executing the atomic services. The first extreme is to have an execution plan that requires a sequential order of execution for all of its sequences. As shown in Figure 12.2-A, the execution

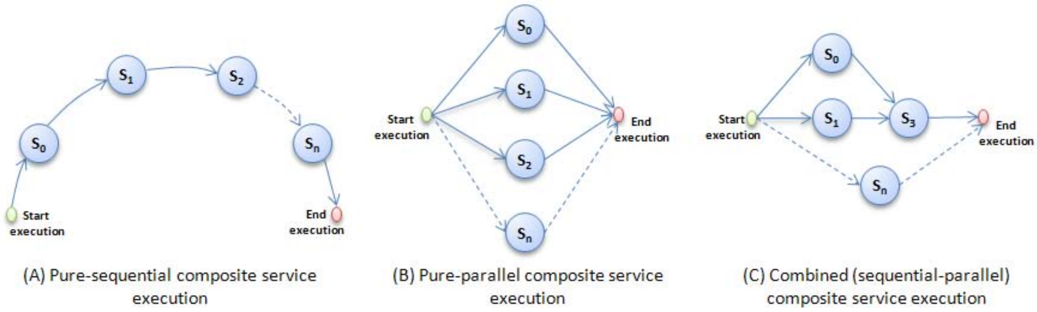


Figure 12.2.: Different execution flows' types of composite service requests

plan requires the results of a service invocation to be fed to the next service and so on until the whole plan is executed. In this case, some of the performance metrics will be expected to reflect low values like the composite service response time. On the other hand, the other extreme is to have a composite service execution plan which requires all of its sequences to be executed in parallel order as in Figure 12.2-B. In this case, although the expected service response time of the composite service request should be better than the first extreme, the probability to have all the required atomic services available in the same network partition becomes lower. therefore, some performance metrics like the composite service availability and success ratio show very low values.
Figure 12.2-C presents a combined (sequential-parallel) composite service execution plan. The expected performance metrics of such a plan will fall in the two performance extremes as it is going to be drawn in the next sections of this chapter.

12.2.2. Managing the Composite Service Execution

The proposed SDP based composite service execution approach applies a centralized managing style for the execution process. There is no need for the clients to pose the composite requests to the providers or other participants to manage the composite service execution process. The mobile nodes (clients) utilize their service discovery information about their local partitions and the available reachable services. As in Figure 12.1-B, the execution process is divided into execution steps. In each of these steps, a service invocation takes place for one of the atomic services inside the sequences of the composite service execution plan. The client takes the responsibility of forwarding the intermediate results between the atomic requests for only itself.

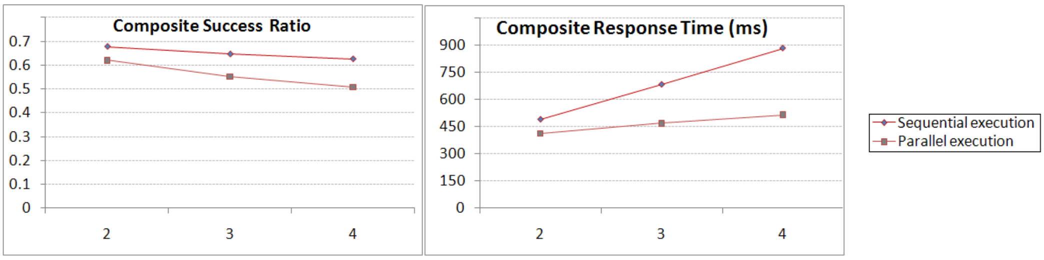


Figure 12.3.: The composite success ratio and response time of both sequential and parallel executions (Y-axis) vs. the composite service request lengths (L)

12.3. Evaluation

The evaluation in this section is done based on simulation. Using the Opnet® Modeler® Wireless simulator, the SDP based composite service execution approach has been implemented. A project file of this simulation is available online[1]. More details about this implementation in Opnet® are available in Appendix C.

12.3.1. Simulation Settings

Network settings

The network participants (50 nodes) are located in a $600m \times 600m$ square area. Each of these nodes can cover a circular transmission range with a radius of $75m$. The Random Waypoint mobility model is applied with speed range $[0, 6]m/s$ and pause time range $[0, 15]$ seconds. The mobile nodes in this simulation are specified as in Section 11.3. The shown results are collected from the averages of 5 runs (each run simulates a 2-hours-network operation time) with different seeds (seed sequence: Starts from:13213, to: 18000 and step: 1000).

Services' settings

The network in the following experiments contains four deployed services that operate side by side without consideration for either the providers' computation power or resources' competence (as in Chapter 11). The four services are deployed on the first available mobile nodes in the network. The requests regarding the services can be categorized into two levels: (a) the requests for the atomic services which are used in the SDP service replication operations (obeys the rich requesting behavior of Chapter 6 for 2 client groups) and (b) the composite service requests. The two

[1] `https://enterprise1.opnet.com/tsts/4dcgi/MODELS_FullDescription?ModelID=951`

levels are distinct and the dissolved atomic requests which are resulting from the composite request are not involved in the service replication, hibernation, caching, and resorting processes of SDP.
The required time to process a request and produce a service response is uniformly distributed between 100 and 200 milliseconds. The length of the composite service request (L) varies between 2 to 4 atomic services.
The composite request has two forms, either all of the contained request sequences are in sequential order (sequential execution) or all of them are required to be executed in a parallel order (parallel execution), as mentioned in Section 12.2.2. Each of the network participants is asked to perform both sequential and parallel requests during the operation of the network. The composite service request generation period is uniformly selected between 300 to 600 seconds to the current simulation time. Appendix C presents more details about the composite request generation process.

Performance metrics

- Composite success ratio:
  The ratio between the correctly served and succeeded composite service requests to the total number of the generated composite service requests regarding a specified length of the composite service request (L).

- Composite response time:
  The average round trip time for the succeeded composite requests between posing the composite service request and receiving the complete response of this composite request. For the sequential composite requests, like in Figure 12.2-A, it is the accumulation of the whole response times of the atomic services. For the parallel composite requests, like in Figure 12.2-B, it should be the maximum response time of the whole response times of the atomic services.

The performance of any form of execution plans, as in Figure 12.2-C, is located in between (in terms of composite success ratio and response time) the two extreme performance indications of the sequential and parallel execution plans of Figure 12.2A and B for the same composite service request lengths.

12.3.2. Performance Analysis

Figure 12.3 presents the composite success ratio and response time of both sequential and parallel executions for different composite service request lengths. Regarding the composite success ratio and for the sequential execution, as the length of the composite request increases, the success ratio slightly decreases. On the other hand, the parallel execution is affected more by increasing the length of the composite request. The proposed approach can achieve a composite success ratio about 0.70 in the case of two sequential atomic services in the composite requests, while the same ratio is less than 0.65 for the same parallel atomic services requests. Since the client checks the availability of all of the atomic services together in the parallel execution before it starts invocation of the composite service requests, the probability for finding all the atomic services active is lower than in the sequential execution where, the client checks only the availability of the atomic services in order and invocation after invocation. Of course, this increases the probability to find available service providers even if they are not available together at a specified time as required in the parallel execution. The decrease of the success ratio in the sequential execution is less than -0.08 while, it is -0.12 in the parallel execution.

For the observed response time values, the sequential response time increases as the length of the composite request increases. The parallel response time increases slightly. The increasing response time in the sequential execution is expected based on the fact that the mobile nodes (clients) wait between the successive atomic requests until responses arrive. The minor increasing in the response time with respect to the length of the composite service is resulting from selecting the maximum response time of the arrived atomic responses at the composite service clients. As the length of the composite service request increases, the possibility to encounter longer response time interval increases. So, the parallel response time increases although, all of its requests for the atomic services are triggered simultaneously. In the cases of sequential executions, the average response time starts about 488 milliseconds with 98 milliseconds as an average standard deviation for composite service length of 2. For the length of 4 services, the average response time becomes 883 milliseconds with 152 milliseconds as an average standard deviation. The increasing rate (with respect to the length of the composite request) in the average response time is about 197 milliseconds per each additional service request (in the composite request). Although, the expected value of the average response time of a single service is 150 milliseconds, the higher observed average response time can be due to the other network processes like routing and transportation.

12.3.3. Extended Performance Analysis

Expressing the required QoS (Quality-of-Service) schemes in terms of time measurements like delay is quite common in computer networks where the quality can be mapped into time constraints [CS99]. In this subsection, the service oriented applications at the mobile clients do not only require proper composite requests to be served but also draw a specified execution time. The performance of the proposed composite service execution approach under a set of time constraints is being presented and investigated in terms of the success ratio and response time.

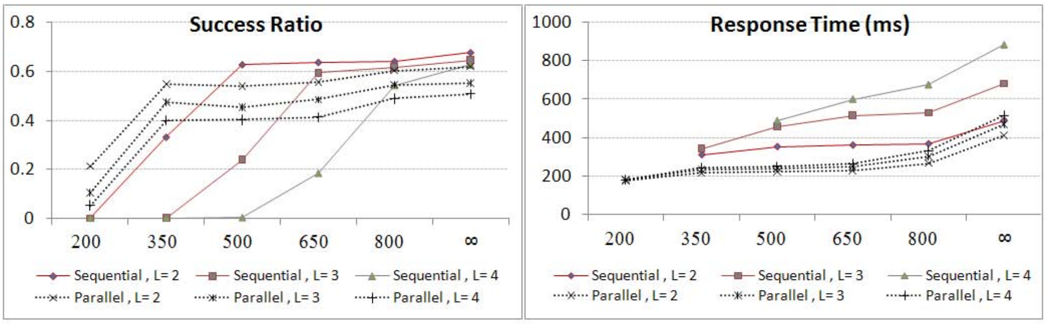


Figure 12.4.: Time constrained performance analysis, success ratio and response time(Y-axis) and the time constraint in milliseconds (X-axis)

For the different lengths of the composite service requests, for both of sequential and parallel execution extremes, each of the clients draws a set of time constraints which restrict the validity of the execution process about the gained results of the posed composite requests. Six scenarios of the enforced time constraints have been added to both of the sequential and parallel execution styles and for each composite request length. In the first scenario, the execution time constraint is 200 milliseconds which means that the execution process (in terms of response time) can not exceed this limit. In the second scenario, the execution time constraint is 350 milliseconds. The execution time constraints for the 3^{rd}, 4^{th} and 5^{th} scenarios are 500, 650 and 800 milliseconds respectively. The sixth scenario contains no time constraints for the execution process.

As shown in Figure 12.4, regarding the average success ratio, of the sequential execution, all the composite service request lengths can not achieve any success ratio for the execution time constraint of 200 milliseconds. Since, the atomic services require about 150 milliseconds (the expected value of the uniform distribution of the service process time, see Subsection 12.3.1) to produce a response for each request, non of the composite request lengths can be served in the first time-constrained scenario. Accordingly, the average response time in this scenario can not be observed.

For the parallel execution style, the three different composite request lengths can be served rarely. Although, the expected execution time of a parallel composite request is about 150 milliseconds (for all parallel composite lengths), a few of these composite requests are served in about 200 milliseconds because of the other required time by the other network processes. As the length of the parallel composite request increases, slightly higher success ratio (about 0.2 for L=2) can be achieved. The observed average response time value for all the succeeded composite requests in the 1^{st} time-constrained scenario is about 200 milliseconds.

In the 2^{nd} time-constrained scenario, for the sequential execution, the observed values of the success ratio can be explained based on the expected value of the execution time of an atomic service requests (150 milliseconds). Since composite services of two atomic service requires roughly about 300 milliseconds to be executed, only those composite service (with lengths equals 2) have the opportunity to be executed. The success ratio in this case is about 0.35, while the average response time is about 350 milliseconds. For the parallel execution in the same time-constrained scenario, all the lengths of composite requests have now a better opportunity to be executed since the longer expected time for the services to be executed is 150 milliseconds. As the time restriction increases, the success ratio of all lengths increases.
For the rest of the time-constrained scenarios, the success ratio of the sequential composite request increases incrementally as the time restriction increases and allows the atomic services to be executed. Starting from time constraint of 800 milliseconds (which we think, it remains very tight taking into consideration the composite service length and the expected execution time of a single atomic service), the achieved success ratios for all composite request lengths are very relative to those achieved ratios where no time constraints are applied in the 6^{th} time-constrained scenario. Regarding the response time, as the time constraints become more relaxing, the obtained average response time values become longer. Moreover, as the composite request length increases, longer average response time values are achieved. The difference between the observed average response time values in the 800-milliseconds scenario and the last scenario with no time constraints is about 100 milliseconds in average independent of the composite request length. This 100 milliseconds time interval draws the border between the requester requirement of having the highest performance of the proposed SDP based composite service execution approach (in terms of success ratio) and enforcing some time constrained execution scheme.

12.4. Topology-Based vs. Inerest-Based Composite Service Execution in MANETs

The protocols in [SZC+08, CJWS08, CWSH08] introduce a set of composite service execution approaches that are based on shorter execution path selection. In [CJWS08], the authors propose a heuristic and dynamic execution path selection based on selecting the lowest hop count from different identical atomic service providers. In a decentralized execution style, each of the intermediate atomic services' provider selects the next intermediate service provider. The last service provider sends the execution results to the original requester node. To estimate the hop count, the composite service execution approach in [CJWS08] assumes the use of the components of a service discovery approach like in [VP05, VRdL05] to get the information of the intermediate provider based on specified routing approach. So, this composite service execution approach lend itself to the area of the network topology analysis.

SDP keeps the minimal number of replicas of the same service inside the same network partition based on the aggregated service popularity. It is trivial in MANETs to associate between the low performance metrics and the long length of the paths in hops. The main reason of association is the limited resources of the intermediate nodes of these paths. As the number of the intermediate nodes increases, the possibility to have failures and insufficient resources on these paths increases as well. If the popularity of the most popular (set) of service(s) is considered as an indication for how this service is centered in its partition and is reachable easily by its clients. Then no need (from SDP based composite service execution module perspective) to estimate it or to be involved in any topological analysis to find out the near providers.

So, on the level of the composite service execution, the important question here is how to state a performance comparison between the two main categories of the composite service execution in MANETs: The interest-based versus the Topology-based composite service execution.

12.4.1. Execution Path

The heuristic dynamic execution path selection approach (HDEP) of [CJWS08] presents a decentralized sequential execution approach. The key function in this approach is choosing the appropriate service provider for each atomic service, which then optimizes the total response time. For each atomic service, the service provider with the lowest hop count is selected based on the selection algorithm of [CJWS08], which is a dynamic and heuristic service selection method that always ensures finding and selecting the "best" service in a neighborhood of set of service provider based on the routing component (AODV [PBRD03]) and in terms of link quality measurements. The intermediate service provider selects the next intermediate service provider dynamically. At the last service provider, it does not select any service provider, it only sends the final execution result directly to the original client. So, the execution paths are dynamically established and optimized to the minimum hop path between the intermediate atomic service providers.

12.4.2. Service Density

The simulation study of [CJWS08] assumes a certain service density in the network that indicates the ratio of the number of same atomic service to the number of nodes. This approach fixes this density during the proposed experiments. From

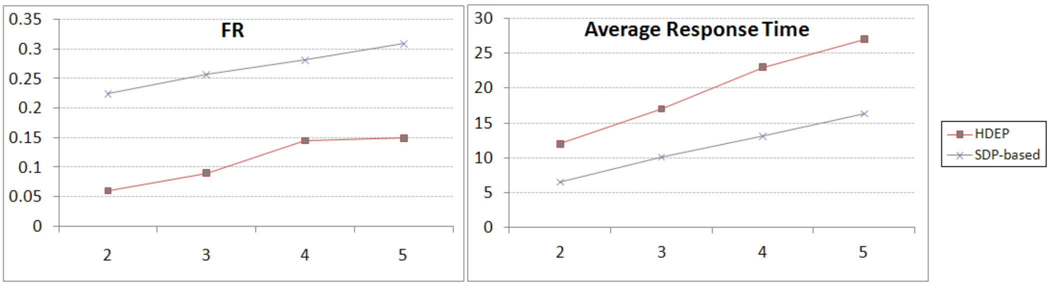


Figure 12.5.: FR and the average response time in seconds of both SDP based execution approach and HDEP (Y-axis) vs. the composite service request lengths(L) (X-axis)

SDP perspective, the service density is equivalent to the observed service prevalence. While the SDP prevalence ratio is obtained to be as minimum as possible in order to save the network resources, HDEP keeps the SD to be 10% . The proposed SDP based composite service execution approach does not require any consideration for the service density during its operations. Moreover, the minimum achieved service prevalence by the employed SDP is an advantage. Even, in cases of an excess of atomic service requests from the perspective of the active service provider's ability to serve, the SDP load balancing mechanism keeps the prevalence as small as possible.

12.4.3. Performance Comparisons

Settings

The stated settings of [CJWS08] have been applied to the Opnet® simulation tool[2] in Appendix C. The network consists of 50 nodes and is placed in a $300m \times 300m$ rectangular area. Each of the mobile nodes can cover a radius of $50m$ transmission range. The random waypoint mobility model is used with maximum velocity of 4 m/s. The service process time is fixed for all of the atomic services to be 3 seconds. One hour of operation time has been given to the network. For SDP, the rich Gross Interest scenario of Chapter 6 and the same composite service requesting behavior as in Section 12.3 are applied.

Performance Metrics

- FR (Failure Rate): The failure rate is the complement value of the success ratio, failure rate = 1 - success ratio.

[2] `https://enterprise1.opnet.com/tsts/4dcgi/MODELS_FullDescription?ModelID=951`

- Average response time.

Performance Analysis

As shown in Figure 12.5, the SDP based approach encounters higher values of FR. In average, it achieves about 0.15 FR with 0.01 standard deviation higher than the HDEP. The lower FR values for HDEP should be considered with the assumed guaranteed service density for each of the atomic services. On the other hand, in the SDP based approach, the same abundance is not guaranteed. For the average response time, the results show how the proposed SDP based approach can reduce the response time in average by about 11.4 seconds with average standard deviation of 5 seconds. In terms of response time, although HDEP is built to optimize the execution path in order to have the minimum hop count between the several service providers, it can not guarantee providing the best execution time. Although, the SDP based approach does not consider the issue of optimizing the execution path (since it is not topology based and apart from its ability to include measurements that can indicate this issue in service/replica popularity ranking), it can provide much better execution time. Of course, this advantage can be due to avoiding inquiring the lower network layer components such as routing component and the related required analysis processes.

12.5. Summary

This chapter shows the relation between service replication and composite service execution. It investigates the feasibility of a SDP based approach for the composite service execution. Background for the composite service execution in MANETs is introduced and a set of different approaches and their concepts are presented and discussed.
SDP is employed in this chapter for composite service execution. As a lower layer for increasing the atomic service availability, SDP plays an important role for achieving the execution process of the composite service requests. An SDP based centralized composite service execution approach is proposed. In this approach, the requesting node (client) takes the controls of the execution process. There is no need for brokers or other third parties to be involved. Only the client and the providers are involved in the proposed execution approach. A detailed simulation study is elaborated to investigate the performance of the proposed SDP based approach. The results are presented and evaluated. Moreover, the evaluations are extended to include the dimension of QoS for the execution process in terms of time constraints. Again, the results are presented and discussed.
In the second part of this chapter, a selected distributed composite service execution approach which is based on topological analysis and tries to optimize the

execution path is introduced. A comparison between this approach and the SDP based approach is presented. The simulation showed higher failure rates for the proposed SDP based approach. On the other hand, the approach showed better response times for the execution process.

The work of this chapter touched on the problem of presence of the composite service requests. By the end of this work, a set of further research questions popped up like: How to define a service popularity for a composite service? How to enhance the success ratio and the failure rates of the proposed SDP based approach? What are the effects of the load balancing mechanisms and the resource restrictions of Chapter 11 on the proposed execution approach?. These questions represent a subset of our future work as introduced and discussed in the next chapter.

Part III.

Finale

CHAPTER 13

Research Goals and Achievements

"Only the bride's family who praises her "

(An Arabic proverb)

In the following sections, the research goals are matched versus the distributed achievements through this thesis. Three research goals have been introduced in Chapter 1. We present each of these goals, then we highlight how the achievements, results and contributions of this thesis covered this research goal.

13.1. Research Goal 1

Research Goal 1 has been stated in Chapter 1 as follows:

To identify challenges, different approaches, and criteria of the service replication in MANETs and cover the related issues. The activities to be considered to meet this goal are:

- Defining the related issues of the service replication in MANETs.
- Surveying the service replication approaches.

This goal is mainly accomplished by the work of Chapter 2 and 3 as follows:
Achievements and results:

- An extended discussion for the related issues that affect service availability in MANETs. Based on this discussion, we introduced our classification for the service replication approaches.
- A survey, based on our classification, for the different service replication approaches in MANETS are presented in Chapter 3.
- In Chapter 5, definitions for service replication as a process and the replicability of services in MANETs are introduced and discussed.
- A comparison is made in Chapter 3 to highlight the differences between a set of service replication approaches for MANETs. In this comparison, we showed how these different approach provide their replication decisions and how all of they (except SDP) are dependent on the network architecture and require network status analyses.
- As an important result, decoupling the replication decisions from the network architecture has been identified to be our conviction for achieving a better replication process.
- Other interesting results achieved in this goal can be mentioned as follows:

- In Chapter 4, The service availability is about 90% in average starting from low-to-moderate network density for both replication and hibernation mechanisms with respect to the mobility.
- The low service prevalence in Chapter 4 which is about 17% in average for all of the introduced network sizes for both replication and hibernation mechanisms with respect to the mobility.
- The low service prevalence in Chapter 5. It is always below 20% for the results of applying both replication and hibernation mechanisms. Moreover, applying both mechanisms with more restrictive replication thresholds ensures a better service prevalence compared to DAR [AYS+09].

13.2. Research Goal 2

Research Goal 2 has been stated in Chapter 1 as follows:

To develop a replication approach that can avoid the coupling between the replication decisions and the network status or components. The activities to be considered to meet this goal are:

- Estimating new sources which can deliver replication decisions.
- Developing replication approaches based on the proposed replication decisions.
- Evaluating the efficiency of the proposed approaches in different scenarios and establishing comparisons to the other replication protocols.

This goal is accomplished as follows:

1. Employing SOA as a standard for managing services in MANETs is introduced in Chapters 2 and 3. In Chapter 4, SDP as a replication protocol for SOA based services is demonstrated. Making replication decisions based on the information about the service popularity is introduced and the SDP performance is evaluated from many aspects on a simplified network model. In Chapter 5, an extended network model is presented for SDP evaluations. Not only the service availability and the prevalence ratio (replication cost) but also the optimality of the service distributions of SDP are evaluated to highlight the efficiency.
   Achievements and results:

- Exploiting of the presence of SOAs in MANETs as a set of standards to achieve service replication is introduced and investigated. In presence of these standards, the proposed replication decisions are suggested to be based the service popularity information which is available in the application layer only. The main achievement here is proving the possibility to make service replication dictions independent on the network status as presented in Chapter 4.
- Investigating the ability of integrating SDP to SOA models, namely the DIANE model [dia], and stating how SDP can use the SOA components is presented and discussed in Chapter 4.
- One of the main contributions in this thesis is introducing the concepts of service popularity and modeling of the client requesting behavior with a quantification method which can describe the popularity based on the Gross Internet specifications.
- Regarding evaluation of SDP, as one of the main contributions in the thesis, the concepts of optimality of the service distributions are introduced in Chapters 5 and 7. In addition to service availability, success ratio, prevalence,replication degree, and residence time, the service distribution optimality of SDP has shown a very consistent and good values through the different experiments. As an example for some of our best results achieved for this point, the experiments showed that, the prevalence ratio SDP is quite low with high service availability and optimality of distributions. As presented in Chapter 8, for a popular service, the service availability is about 88% in average with a success ratio about 78% in average, while the prevalence ratio is about 10% in average. Moreover, in terms of optimality, SDP can achieve at most 1.41 active replicas in the Typical Partition.

2. Four performance comparisons between SDP and the other approaches are presented in this thesis. The fist comparison between SDP and DAR [AYS+09] in Chapter 5 highlights the differences in their performance from an energy consumption perspective. In the second part of Chapter 9, the second performance comparison between SDP and two partitioning-aware replication protocols, PSRP and SSRP [DB07, DB08a], is held. Then, the third comparison is held to show the performance of SDP considering two proposed service selection modes to PSRP and SSRP [DB07, DB08a] in Chapter 10. In Chapter 12, the last comparison between a proposed SDP based composite service execution approach to a heuristic decentralized topology aware composite service execution approach HDEP [CJWS08]. Not only comparing the SDP performance to the other approaches and protocols but also investigat-

ing different mobility specifications and models are included by our work. As mentioned in Chapter 5, mobility plays an important role in the service prevalence. In Chapter 4 and 5, the Random Waypoint mobility model with long pause time intervals is introduced. In Chapter 9, the area graph based mobility model is used with attributed heterogeneity of mobility settings. In the second part of Chapter 9, SDP applied with more mobility specifications for the Random Waypoint mobility model.

Achievements and results:

- In Chapter 5, although the comparison applied SDP with less-meaningful specifications for replication and hibernation mechanisms, the results reflect how SDP performance is close and comparable to DAR as a network status approach protocols.
- After describing the meanings of the different SDP specifications and their effects in Chapter 7, a comparison between SDP to both PSRP and SSRP is made in Chapter 9 which showed the differences in performance between SDP and both PSRP and SSRP as partitioning-aware protocols. Some of the important results to be mentioned are: Starting from moderate network densities (50 to 60 nodes) SDP can achieve a better service availability for the popular services (about 90% in average in a set of high mobility specifications) than PSRP. On the other hand, starting from low network densities (40 nodes), SDP can achieve a better service distribution optimality than SSRP and starting from 50 nodes, it is also better than PSRP and about (80% in average). For more details, see Chapter 9.
- By introducing new concepts for SDP service selection, the "wakeup-neighbors" mode in Chapter 10, the comparison of SDP to both PSRP and SSRP showed that SDP starting from low network densities (about 30 nodes) can achieve better availability (about 90% in average) compared to PSRP and optimality about 80% in average compared to SSRP for a popular service. For more details, see Chapter 10 .
- Comparing SDP to HDEP for composite service execution showed that SDP has a higher response time in general than HDEP but on the other hand it has some deficiencies in the success ratio of the composite request. For more details, see Chapter 12.
- Applying Different mobility specifications and models is a hot topic in this work, since the mobility has an important role in the service prevalence as assumed in Chapter 4. By applying a heterogeneous realistic mobility specifications, Area graph Based Mobility Model (AGM) [BRS05], SDP keeps a reasonable performance as discussed in Chapter 9. The

important result is that the density of the formed partition has the main role in generating more optimal service distributions. This result supports our consideration about the role of the network size in the service popularity as discussed in Chapter 6.

13.3. Research Goal 3

Research Goal 3 has been stated in Chapter 1 as follows:

To consider and investigate the resource awareness and composite service execution impacts by the proposed replication approach. The activities to be considered to meet this goal are:

- Developing of resource conservative scenarios.
- Investigating the effects of these scenarios on the proposed service replication approach.
- Introducing the problems of composite service execution in MANETs and the relation to service replication.
- Estimating the performance of the composite service execution with ad of the proposed service replication approach.

This goal is accomplished by the work of Chapter 11 and 12 as follows:

1. In Chapter 11, the resource awareness issues are considered from two perspectives. First, the availability of the resources by the provider side and how can the service responses be affected. Second, The availability of the resources at the clients. The scenarios of resources awareness are built form these two perspectives and SDP is investigated with more challenging environments.
   Achievements and results:
   - One of the interesting results achieved in Chapter 11 is the success ratio achieved in the experiments of the competing services. The observed decrease in success ratio, compared to scenarios without competition, is limited to about 12% in average.
   - About the optimality in terms of WRCR, a measurement for the correctness of service distributions (see Chapter 7), it remains with similar values, about 72% in average, for the competing services compared to scenarios without competition.

2. Chapter 12 introduces the problems and issues of the composite service execution in MANETs. An SDP based composite service execution approach is introduced and compared to another approach.
   Achievements and results:
   - A set of time constrained scenarios for composite requests is introduced for a mimic of time based QoS constraints. For varying lengths of composite requests, between 2 to 4 atomic service requests, both success ratio and response time have been investigated against $[200, \infty)$ milliseconds constraints.
   - A comparison between a heuristic dynamic execution path selection approach (HDEP) [CJWS08] and the SDP based approach for composite service execution is presented. The SDP approach showed better response times for the execution process. In average, the SDP based approach can achieve about 65% reduction in the response time.

13.4. Summary

In this chapter, we presented a demonstration for the contributions achieved by the work of this thesis and how they matches our initial set of research goals. Moreover, we re-highlighted the most important results in a flow with the matched goals to reflect how these results have been used in the progress of the work.

CHAPTER 14

Conclusions and Future Work

"Drawing is the honesty of the art. There is no possibility of cheating. It is either good or bad"

(Salvador Dali)

This chapter summarizes the work of this thesis. It repeats briefly the important results and highlights the main contributions of the thesis. The structure of this chapter is as follows: In Section 14.1, a summary for the work of this thesis is presented. Then, Possible future work based on this work is introduced in Section 14.2

14.1. Summary

The work of this thesis is divided into three parts. Part I includes an introduction chapter, Chapter 1, which presents the areas related to this thesis, our motivations and research goals, the achievements and contributions which are introduced by this work, the evaluation methods and flows during the next chapters, and the structure of the thesis.
Chapter 2 highlights briefly the related background of this thesis. MANETs and their features are presented. The need of sharing resources as services is discussed. Deploying SOAs and the related challenges are demonstrated. Replication is introduced as a solution techniques for increasing the service availability in MANETs. The other topics which are involved by this work like MANETs' performance evaluation, the leader election problem, the load balancing, QoS and resource awareness, and composite service execution are mentioned.
Chapter 3 presents the state of the art for data and service replication in MANETs. It defines the common terminology to be used in the rest of the chapters of this work. It introduces the service replication as a data replication approach with special features which are summarized as the "Service Awareness". The issues that influence the replication processes in MANETs are demonstrated. The set of parameters that can describe the replication performance of different approaches are introduced. Our discussion showed how these parameters can be categorized.
Part II includes the chapters that introduce, propose, build and evaluate SDP concepts, mechanisms, features, and components. In Chapter 4, the main concepts behind SDP are introduced and tested against a simple network model where the network does not get partitioned. The important results of Chapter 4 is the feasibility of applying the proposed SDP in a simple dynamic network model. The integration between SDP and SOAs is presented. The interactions between the DIANE model[1] components are demonstrated. The initial general assumptions for SDP are made in Chapter 4.
Chapter 5 includes an extensive network model. One of the contributions of this thesis is introducing definitions for the service replication process and the replicable services based on the presented network model. The main mechanisms of SDP are extended and polished. Moreover, the SDP considerations for the service

[1] As a SOA discovery model for MANETs [dia]

synchronization and mobility are demonstrated. An extended evaluation for the performance of SDP with respect to the proposed extended network model is presented. Another important contribution of Chapter 5 is introducing the concepts of the optimality of the generated service distributions. Two computations methods (ratios) are proposed to indicate the service distribution optimality: the Linear Correctness Ratio (LCR) and the Rational Correctness (RCR). These ratios are based on the assumptions of Chapter 4 where one replica is assumed to satisfy all the service requests of its network partition. Chapter 5 ends with a presentation of a comparison study between the proposed SDP and another service replication protocol, DAR [AYS+09], to determine SDP's performance relative to existing approaches at this early stage of the progress of the work of this thesis.
In Chapter 6, more considerations for the client requesting behavior regarding a specific service and the effects on the service popularity are introduced. The main contributions of Chapter 6 are a general client requesting model that can capture each client requesting behavior and aggregating the generated requests to produce an indication for the service popularity which is the Gross Interest. The effects of the different criteria of the proposed generalized requesting model on the resultant Gross Interest are demonstrated. Two main Gross Interest scenarios are configured with different specifications for the requesting rates to model different service popularity and to be used in the rest of the work of this thesis.
The goals behind Chapter 7 is to find out the effects of different specifications for the replication and hibernation settings on the general performance of SDP. It introduces extended considerations for the resultant service distributions and evaluates the SDP allocation correctness in different replication and hibernation behavior. The main contribution of Chapter 7 is finding the Weighted Rational Correctness Ratio (WRCR) which is used in the rest of the chapters as an indication for the optimality of the service distribution process. The important result of Chapter 7 is the positive influence of the short time interval of the service hibernation test on the optimality of the service distributions.
Chapter 8 extends the investigations for the efficient length of time interval of the hibernation tests by the service providers. Two leader election modes are introduced to manage the hibernation mechanism by SDP. The short election hibernation mode enforces the providers to test their replica after an assumed minimum required time for a service offer publishing. The important result of Chapter 8 is how applying the short election mode can ensure correct service distributions with higher SDP service availability and success ratio.
In Chapter 9, SDP performance is evaluated with more realistic mobility models. In the first part of Chapter 9, a set of mobility models based on the area graph based mobility model [BRS05] is proposed. The heterogeneity of the proposed specifications for the models is formed from two levels: the Gross Interest and the mobility settings in the different proposed places. An important contribution in

this part of Chapter 9 is the quantification the average velocity of the mobile nodes inside each of the models' places. The important conclusion of this chapter is the steady and high performance of SDP. In the second part of Chapter 9, SDP performance is compared to two network status aware protocols, namely PSRP and SSRP [DB07, DB08a]. This comparison is more mature than the comparison of Chapter 5 to DAR. SDP performance key specifications are optimized. Moreover, two mobility sets of specifications are presented. A special consideration for the service general availability and the success ratio is demonstrated and discussed. The optimality of the service distributions generated by the three protocols is investigated.
In Chapter 10, new service selection strategies are investigated as a source for enhancing both service availability and distribution optimality. Two service selection modes are introduced. The better SDP performance of the "wakeup-neighbors" mode in terms of the availability (success ratio) and WRCR are shown and investigated. The important contribution of this chapter is the proposed mechanism of the "wakeup-neighbors" mode that enables the interested client to wakeup a hibernated service provider if it is needed. By this mode, the availability of the cached replicas is generalized for the neighboring nodes.
Chapter 11 presents two levels of resource restrictions and the required awareness thereof by SDP. The first level is the restricted resources at the service providers and the second level is the resources at the interested clients. Chapter 11 uses the contribution of the generalized availability of Chapter 10 and provides an interest based load balancing mechanism to distribute dynamically the load of the requests over many service providers and to avoid the request congestion in cases of less resourced service providers. The assumption of Chapter 4 that assumes "one service per network partition" will satisfy all generated requests is deprecated in Chapter 11. It considers that not all of the service providers can serve alone all the requests of the clients regarding their services. The ability of serving new requests is assumed to be based on the available resources at these providers and the currently connected sessions to these services. Effects of the presence of many active service providers in the same network partition violate the basic assumptions of the allocation correctness ratios of Chapter 5 and 7. Therefore, a new allocation correctness ratio that considers the proposed load balancing mechanism is proposed. The assumptions of the available resources on the mobile nodes which enable them to participate in the replication and caching processes of Chapter 4 are also deprecated in Chapter 10. A scenario of competing services is used to investigate the performance of SDP where not all of the mobile nodes can participate in the replication and caching processes. Moreover, the proposed load balancing mechanism is applied within the scenario of the competing services.

In Chapter 12, the composite service execution is introduced as a challenge for SDP. As stated in our motivations, increasing the availability of atomic services may ensure a better composite service execution process. In this chapter, we investigate how can SDP support the composite service execution. An introduction and background for the composite service execution in MANETs is demonstrated. Then, a SDP based composite service execution approach is proposed. The performance of the proposed approach is investigated in terms of response time and success ratio versus a variant length of the composite service requests. The evaluations are extended to include a set of time constraints' scenarios for achieving the execution process. These scenarios are a mimic for the QoS time based constraints. The important result here is determining the limits of the proposed approach performance between the full sequential and the full parallel composite service flows. Moreover, the proposed approach is compared to another composite service execution approach (HDEP [CJWS08]) for MANETs. This comparison is stated as a comparison between an interest based and topology based approaches for composite service execution. The results are investigated and the advantages and disadvantages of the proposed approach are stated and discussed.

14.2. Future Work

The suggested future work by the end of this thesis can be based on the flow of the chapters of Part II.
Based on the contributions and results of Chapter 4, we consider the semantic support for SDP for future work. Including semantics for SDP may introduce a better service replication processes for Web services. Semantic layers for supporting better popularity rating, service selection, and execution are required to be investigated.
In Chapter 5, SDP is compared to DAR which is a network status and energy consumption aware service replication protocol. Energy is a vital and very limited resource in mobile nodes. We consider for the future work investigations for enabling SDP to optimize the energy consumed by its operations.
Chapter 6 describes the basic considerations of the service popularity. Combining other functional and non functional criteria like the request cost, trust, sizes of buffers allowed by the servers, response time, and loyalty of the customers (clients) to be considered in computing the Requirements' Index for better service popularity indications is suggested for the future work. Moreover, we also suggest searching for other methods for the aggregation of service popularity and considering the issues of the new deployed services where the service popularity is not yet aggregated. Issues of the semantic support for the proposed service popularity concepts and entities should be investigated. Modeling real service or profiles for groups of services for the proposed generalized client requesting model in Chapter 6 represents a hot topic for our future work.
Chapters 7, 8 and 10 present the main behavior of SDP mechanisms and their

influencing parameters. The replication, hibernation and caching behaviors are based only on the number of client requests. Other parameters like deploying new versions of the service and the cache invalidation behaviors should be taken into consideration for the future work.
Chapter 11 introduces the concept of unavailable resources by the clients to participate in the replication process. The presence of these passive clients needs more investigation about their ratio to the total network size and their interest in the service.
In Chapter 12, although achieving many advantages, SDP has higher failure rates compared to others. Finding methods for increasing the composite success ratios and decreasing these failure rates are considered also as future work.

References

[AB08] Ebtisam Amar and Selma Boumerdassi. A location service for position-based routing in mobile ad hoc networks. In *Proceedings of the 8th international conference on New technologies in distributed systems (NOTERE 08)*, pages 1–4, New York, NY, USA, 2008. ACM.

[AH74] Alfred V. Aho and John E. Hopcroft. *The Design and Analysis of Computer Algorithms*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1974.

[AYS+09] Asaad Ahmed, Keiichi Yasumoto, Naoki Shibata, Tomoya Kitani, and Minoru Ito. DAR: Distributed adaptive service replication for manets. In *Proceedings of the 5th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob 2009)*, pages 91–97, Marrakech, Morocco, 2009.

[BBHS05] Joos-Hendrik Böse, Frank Bregulla, Katharina Hahn, and Manuel Scholz. Adaptive data dissemination in mobile ad-hoc networks. In *Die 5. Jahrestagung der Gesellschaft für Informatik (INFORMATIK 2005)*, pages 528–532, Bonn, Germany, 2005.

[BCM05] Paolo Bellavista, Antonio Corradi, and Eugenio Magistretti. Redman: An optimistic replication middleware for read-only resources in dense manets. *Pervasive Mobile Computing*, 1(3):279–310, 2005.

[BDS07] Bartosz Biskupski, Jim Dowling, and Jan Sacha. Properties and mechanisms of self-organizing manet and p2p systems. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 2(1):1, 2007.

[BHI10] Jeppe Bronsted, Klaus Marius Hansen, and Mads Ingstrup. Service composition issues in pervasive computing. *IEEE Pervasive Computing*, 9:62–70, 2010.

[BHPC04] Christian Bettstetter, Hannes Hartenstein, and Xavier Pérez-Costa. Stochastic properties of the random waypoint mobility model. *Wireless Networks*, 10(5):555–567, 2004.

[BKAL05] Mohamed Brahma, K. Kim, Abdelhafid Abouaissa, and Pascal Lorenz. A load-balancing and push-out scheme for supporting QoS in manets. *Telecommunication Systems*, 30(1-3):161–175, 2005.

[BLJM08] Norbert Bieberstein, Robert G. Laird, Keith Jones, and Tilak Mitra. *Executing SOA: A Practical Guide for the Service-Oriented Architect*. Addison-Wesley, first edition, 2008.

[BRS05] Sven Bittner, Wolf-Ulrich Raffel, and Manuel Scholz. The area graph-based mobility model and its impact on data dissemination. In *Proceedings of the 3rd IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW 05)*, pages 268–272, Washington, DC, USA, 2005. IEEE Computer Society.

[CC05] Duckki Lee Chunglae Cho. Survey of service discovery architectures for mobile ad hoc networks. *Term paper, Mobile Computing, CEN 5531, Department of Computer and Information Science and Engineering (CICE), University of Florida*, 2005.

[Cha04] Kamalsinh F. Chavda. Anatomy of a web service. *Journal of Computing Sciences in Colleges*, 19(3):124–134, 2004.

[Cha05] Tianying Chang. *Enabling Scalable Information Sharing for Distributed Applications Through Dynamic Replication*. PhD thesis, College of Computing, Georgia Institute of Technology, Dec. 2005.

[Che99] Shigang Chen. *Routing Supagesort for Providing Guaranteed End-to-End Quality-of-Service*. PhD thesis, Engineering College, University of Illinois, 1999.

[CJFY05] Dipanjan Chakraborty, Anupam Joshi, Tim Finin, and Yelena Yesha. Service composition for mobile environments. *Mobile Network Applications*, 10(4):435–451, 2005.

[CJWS08] Weiyu Chen, Yinan Jing, Jingjing Wu, and Weiwei Sun. A dynamic execution path selection approach for composite services in manets. In *Proceedings of the 4th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM 08)*, pages 1–4, Dalian, China, 2008.

[CN01] Kai Chen and Klara Nahrstedt. An integrated data lookup and replication scheme in mobile ad hoc networks. In *Proceedings of the International Symposium on the Convergence of Information Technologies and Communications (SPIE 2001)*, pages 1–8, 2001.

[CPK+08] Miseon Choi, Wonik Park, Young-Kuk Kim, Miseon Choi, Wonik Park, and Young-Kuk Kim. A split synchronizing mobile transaction model. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication (ICUIMC 08)*, pages 196–201, Suwon, Korea, 2008. ACM.

[CS99] Dan Chalmers and Morris Sloman. A survey of quality of service in mobile computing environments. *IEEE communications surveys*, Second Quarter 1999.

[CWSH08] Weiyu Chen, Jingjing Wu, Weiwei Sun, and Zhenying He. A location based execution path selection for composite service in manets. In *Proceedings of the 9th International Conference for Young Computer Scientists (ICYCS 08)*, pages 533–538, Washington, DC, USA, 2008. IEEE Computer Society.

[DB06] Abdelouahid Derhab and Nadjib Badache. Localized hybrid data delivery scheme using k-hop clustering algorithm in ad hoc networks. *IEEE International Conference on Mobile Adhoc and Sensor Systems Conference*, 0:668–673, 2006.

[DB07] Abdelouahid Derhab and Nadjib Badache. A pull-based service replication protocol in mobile ad hoc networks. *European Transactions on Telecommunications*, 18:1–11, 2007.

[DB08a] Abdelouahid Derhab and Nadjib Badache. Self-stabilizing algorithm for high service availability in spite of concurrent topology changes in ad hoc mobile networks. *Journal of Parallel Distributed Computing*, 68(6):752–768, 2008.

[DB08b] Abdelouahid Derhab and Nadjib Badache. A self-stabilizing leader election algorithm in highly dynamic ad hoc mobile networks. *IEEE Transactions on Parallel Distributed Systems*, 19(7):926–939, 2008.

[DB09] Abdelouahid Derhab and Nadjib Badache. Data replication protocols for mobile ad-hoc networks: A survey and taxonomy. *IEEE Communications SurveyS and Tutorials*, 11(2):33–51, 2009. Second Quarter.

[DE08] Orhan Dagdeviren and Kayhan Erciyes. A hierarchical leader election protocol for mobile ad hoc networks. In *Proceedings of the 8th International Conference on Computational Science (ICCS 2008)*, pages 509–518, Kraków, Poland, 2008.

[DHY03] H. Du, H. Hassanein, and C. Yeh. Zone-based routing protocol for high-mobility manet. In *Proceedings of the Canadian Conference on Electrical and Computer Engineering (CCECE 2003)*, 2003.

[dia] Diane - services in ad hoc networks (research project , http://fusion.cs.uni-jena.de/diane/.

[DJ07] Schahram Dustdar and Lukasz Juszczyk. Dynamic replication and synchronization of web services for high availability in mobile ad-hoc networks. *Service Oriented Computing and Applications*, 1:19–33, March 2007.

[DPM02] O. Dousse, P.Thiran, and M.Hasler. Connectivity in ad-hoc and hybrid networks. In *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, 2002.

[EN06] Ramez Elmasri and Shamkant B. Navathe. *Fundamentals of Database Systems (5th Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2006.

[FKM06] R. Favade, V.S. Kaulgud, and S.A. Mondal. An enhanced service discovery protocol for bluetooth scatternets. In *Proceedings of the 2006 ACS/IEEE International Conference on Pervasive Services (PERSER 06)*, pages 261–264, Washington, DC, USA, 2006. IEEE Computer Society.

[Fon04] Michelle W. L. Fong. *E-Collaborations and Virtual Organizations*. IRM Press, 2004.

[FP05] E. Ferro and F. Potorti. Bluetooth and wi-fi wireless protocols: a survey and a comparison. *Wireless Communications, IEEE*, 12(1):12 – 26, feb. 2005.

[Gas05] Matthew S Gast. *802.11 Wireless Networks: The Definitive Guide, Second Edition*. O'Reilly Media, Inc., 2005.

[GB99] Richard Golding and Elizabeth Borowsky. Fault-tolerant replication management in large-scale distributed storage systems. In *Proceedings* *of the 18th IEEE Symposium on Reliable Distributed Systems (SRDS 99)*, page 144, Washington, DC, USA, 1999. IEEE Computer Society.

[GJ79] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York: W.H. Freeman and Co., second edition, 1979.

[GPVD99] E. Guttmann, C. Perkins, J. Veizades, and M. Day. Service location protocol. IETF Internet Draft, RFC 2608, 1999.

[GT95] Mario Gerla and Jack Tzu-Chieh Tsai. Multicluster, mobile, multimedia radio network. *Wireless Networks*, 1(3):255–265, 1995.

[GVDS01] N. Golmie, R. E. Van Dyck, and A. Soltanian. Interference of bluetooth and ieee 802.11: simulation modeling and performance evaluation. In *Proceedings of the 4th ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems (MSWIM 01)*, pages 11–18, Rome, Italy, 2001. ACM.

[Ham04] Mohamed Hamdy. Quality-of-service routing for ad hoc wireless networks. Master's thesis, Department of Information Systems, Faculty of Computer and Information Sciences, Ain Shams University, Cairo, Egypt, February 2004.

[Han05] Radu Handorean. *Context Aware Service Oriented Computing in Mobile Ad Hoc Networks*. PhD thesis, The Henry Edwin Sever Graduate School of Washington University, Department of Computer Science and Engineering, Saint louis, Missouri, December 2005.

[Han06] Radu Handorean. Supporting predictable service provision in manets via context aware session management. *International Journal of Web Services Research (JWSR)*, 3, No.3:1–26, 2006.

[Har01] T. Hara. Effective replica allocation in ad hoc networks for improving data accessibility. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM 2001)*, pages 1568–1576, 2001.

[Har03] Takahiro Hara. Replica allocation methods in ad hoc networks with data update. *Mobile Networks and Applications*, 8:2003, 2003.

[HBG06] Luc Hogie, Pascal Bouvry, and Frédéric Guinand. An overview of manets simulation. *Electronic Notes in Theoretical Computer Science*, 150(1):81–101, 2006.

[HC06] Jiun-Long Huang and Ming-Syan Chen. On the effect of group mobility to data replication in ad hoc networks. *IEEE Transactions on Mobile Computing*, 5(5):492–507, 2006.

[HCP03] Jiun-Long Huang, Ming-Syan Chen, and Wen-Chih Peng. Exploring group mobility for replica data allocation in a mobile environment. In *Proceedings of the 12th international conference on Information and knowledge management (CIKM03)*, pages 161–168, New York, NY, USA, 2003. ACM.

[HDKR10] Mohamed Hamdy, Abdelouahid Derhab, and Birgitta König-Ries. A comparison on manets' service replication schemes: Interest versus topology prediction. In *Proceedings of the 2nd International Conference on Wireless and Mobile Networks (WiMo 2010)*, volume 84 of *Communications in Computer and Information Science CCIS*, pages 202–216, Ankara, Turkey, June 2010. Springer Berlin Heidelberg.

[HGPC99] Xiaoyan Hong, Mario Gerla, Guangyu Pei, and Ching-Chuan Chiang. A group mobility model for ad hoc wireless networks. In *Proceedings of the 2nd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems (MSWiM 99)*, pages 53–60, New York, NY, USA, 1999. ACM.

[HHN05] Hideki Hayashi, Takahiro Hara, and Shojiro Nishio. Updated data dissemination methods for updating old replicas in ad hoc networks. *Personal Ubiquitous Computing*, 9(5):273–283, 2005.

[HHN06] Hideki Hayashi, Takahiro Hara, and Shojiro Nishio. On updated data dissemination exploiting an epidemic model in ad hoc networks. In *Proceedings of the Biologically Inspired Approaches to Advanced Information Technology, Second International Workshop (BioADIT 06)*, pages 306–321, Osaka, Japan, January 2006.

[HKR06] Mohamed Hamdy and Birgitta König-Ries. Service-orientation in mobile computing - an overview. In *Proceedings of the Tools and Applications for Mobile Contents Workshop (TAMC2006) in Conjunction with the 7th International Conference on Mobile Data Management (MDM06)*, page 138, Nara, Japan, 2006. IEEE Computer Society.

[HKR08a] Mohamed Hamdy and Birgitta König-Ries. An extended analysis of an interest-based service distribution protocol for mobile ad hoc networks. In *Proceedings of the International Conference on Wireless Information Networks and Systems (WINSYS 2008)*, Porto, Portugal, 2008.

[HKR08b] Mohamed Hamdy and Birgitta König-Ries. Service availability, success ratio, prevalence, replica allocation correctness, replication degree, and effects of different replication/hibernation behavior effects of the service distribution protocol for mobile ad hoc networks -a detailed study-. Technical report, 01.12.2008 Math/Inf/08/08, Institute of Computer Science, Friedrich-Schiller-University of Jena, 2008.

[HKR08c] Mohamed Hamdy and Birgitta König-Ries. A service distribution protocol for mobile ad hoc networks. In *Proceedings of the International Conference on Pervasive Services (ICPS 08)*, Sorrento, Italy, 2008.

[HKR09a] Mohamed Hamdy and Birgitta König-Ries. *Communications in Computer and Information Science, Book of selected and extended papers of the ICETE 2008*, volume 48 of *CCIS 48*, chapter The Service Distribution Protocol for MANETs- Criteria and Performance Analysis, pages 467–479. Springer Berlin Heidelberg, 2009.

[HKR09b] Mohamed Hamdy and Birgitta König-Ries. A comparison study for service replication protocols in manets. Technical report, 01.06.2009 Math/Inf/02/09, Institute of Computer Science, Friedrich-Schiller-University Jena, Jena, Germany, June 2009.

[HKR09c] Mohamed Hamdy and Birgitta König-Ries. Effects of different hibernation behaviors on the service distribution protocol for mobile networks and its replica placement process. In *Proceedings of the International Workshop on the Role of Services, Ontologies, and Context in Mobile Environments (RoSOC-M '09) in conjunction with the 10th International Conference on Mobile Data Management (MDM09)*, Taipei, Taiwan, May 2009.

[HKR09d] Mohamed Hamdy and Birgitta König-Ries. Leader election modes of the service distribution protocol for mobile ad hoc networks. In *Proceedings of the 4th Conference on Mobility and Mobile Information Systems (4. Konferenz Mobilität und mobile Informationssysteme) (MMS 2009)*, Münster, Germany, March 2009.

[HKR10a] Mohamed Hamdy and Birgitta König-Ries. Impact of heterogeneous mobility models on the service distribution protocol for manets. In *Proceedings of the IADIS International Conference Wireless Applications and Computing (WAC2010)*, Freiburg, Germany, 2010.

[HKR10b] Mohamed Hamdy and Birgitta König-Ries. An interest-based load balancing mechanism for the service distribution protocol in manets. In

Proceedings of the 8th International Conference on Advances in Mobile Computing and Multimedia (MoMM2010), Paris, France, 2010.

[HKR10c] Mohamed Hamdy and Birgitta König-Ries. New service selection strategies for the service distribution protocol for manets. In *Proceedings of the IADIS International Conference Wireless Applications and Computing (WAC2010)*, Freiburg, Germany, 2010.

[HKR10d] Mohamed Hamdy and Birgitta König-Ries. Sdp-implementation in opnet® v.1: Manetsśervice replication and load balancing: Release notes-user manual-programmer guide. Technical report, FRIEDRICH-SCHILLER-UNIVERSITY OF JENA, April 2010.

[HKR10e] Mohamed Hamdy and Birgitta König-Ries. Sdp-implementation in opnet® v.2, sdp-based composite service execution in manets: Release notes-user manual-programmer guide. Technical report, Friedrich-Schiller-University of Jena, June 2010.

[HKR10f] Mohamed Hamdy and Birgitta König-Ries. Sdp primary implementation in microsoft® visualc++®, a service distribution protocol for manets. Technical report, Friedrich-Schiller-University of Jena, April 2010.

[HKR11] Mohamed Hamdy and Birgitta König-Ries. *Handbook of Research on Non-Functional Properties for Service-Oriented Systems: Future Directions*, chapter The Gross Interest: Service Popularity Aggregation. Advances in Knowledge Management (AKM). IGI GLOBAL, Primarily accepted proposal, to appear in 2011.

[HKRK07] Mohamed Hamdy, Birgitta König-Ries, and Ulrich Küster. Non-functional parameters as first class citizens in service description and matchmaking - an integrated approach. In *Proceedings of the International Conference on Service-Oriented Computing (ICSOC 2007) Workshops: The 1st International Workshop on Non Functional Properties and Service Level Agreements in Service Oriented Computing (NFPSLA-SOC2007), Revised Selected Papers*, pages 93–104, Berlin, Heidelberg, 2007. Springer-Verlag.

[HLN03] Takahiro Hara, Yin-Huei Loh, and Shojiro Nishio. Data replication methods based on the stability of radio links in ad hoc networks. In *Proceedings of the International Workshop on Database and Expert Systems Applications*, volume 0, page 969, Los Alamitos, CA, USA, 2003. IEEE Computer Society.

[HLT04] Chi-Fu Huang, Hung-Wei Lee, and Yu-Chee Tseng. A two-tier heterogeneous mobile ad hoc network architecture and its load-balance routing problem. *Mobile Networks Applications*, 9(4):379–391, 2004.

[HM05] Takahiro Hara and Sanjay K. Madria. Consistency management among replicas in peer-to-peer mobile ad hoc networks. In *Proceedings of the 4th IEEE Symposium on Reliable Distributed Systems (SRDS'05)*, pages 3–12, Orlando, FL, USA, 2005.

[HM09] Takahiro Hara and Sanjay Kumar Madria. Consistency management strategies for data replication in mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, 8:950–967, 2009.

[HP04] M. Hauspie and A. Panier. Localized probabilistic and dominating set based algorithm for efficient information dissemination in ad hoc networks. In *Proceedings of the IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS04)*, pages 11 – 20, Florida, USA, October 2004.

[HSC01] Michael Hauspie, David Simplot, and Jean Carle. Replication decision algorithm based on link evaluation for services in manet. Technical report, University of Lille, France, 2001.

[HW01] Zygmunt J. Haas and Abhijit Warkhedi. The design and performance of mobile tcp for wireless networks. *High Speed Networks*, 10(3):187–207, 2001.

[JDT05] Ivar Jorstad, Schahram Dustdar, and Do Van Thanh. Service-oriented architectures and mobile services. In *Proceedings of the 3rd International Workshop on Ubiquitous Mobile Information and Collaboration Systems (UMICS), co-located with (CAiSE 05)*, pages 617–631, Porto, Portugal, 2005.

[JJKY04] Zheng Jing, Su Jinshu, Yang Kan, and Wang Yijie. *Stable Neighbor Based Adaptive Replica Allocation in Mobile Ad Hoc Networks*, chapter Computational Science - ICCS 2004, pages 373–380. Lecture Notes in Computer Science. Springer Berlin - Heidelberg, 2004.

[JM96] David B. Johnson and David A. Maltz. *Dynamic Source Routing in Ad Hoc Wireless Networks*, chapter Mobile Computing: Chapter 5, pages 153–181. Kluwer Academic Publishers, 1996.

[Jos07] Nicolai M. Josuttis. *SOA in Practice: The Art of Distributed System Design*. O'Reilly, first edition, 2007.

[JSHS04] Milenko Jorgic, Ivan Stojmenovic, Michaël Hauspie, and David Simplotryl. Localized algorithms for detection of critical nodes and links for connectivity in ad hoc networks. In *Proceedings of the 3rd Annual IFIP Mediterranean Ad Hoc Networking Workshop, Med-Hoc-Net*, pages 360–371, 2004.

[Jur06] Matjaz B. Juric. *Business Process Execution Language for Web Services BPEL and BPEL4WS 2nd Edition*. Packt Publishing, 2006.

[Jus05] Lukasz Juszczyk. Replication and synchronization of web services in ad-hoc networks. Master's thesis, Technischen Universität Wien, 2005.

[JW05] Shudong Jin and Limin Wang. Content and service replication strategies in multi-hop wireless mesh networks. In *Proceedings of the 8th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems (MSWiM '05)*, pages 79–86, New York, NY, USA, 2005. ACM.

[JXS07] Shanshan Jiang, Yuan Xue, and Schmidt. Minimum disruption service composition and recovery over mobile ad hoc networks. In *Proceedings of the 4th Annual International Conference on Mobile and Ubiquitous Systems: Networking Services (MobiQuitous 07)*, pages 1–8, 2007.

[KÖ7] Günter Küppers. *Simulation*, volume 25 of *Sociology of the Sciences Yearbook*, chapter Computer Simulation: Practice, Epistemology, and Social Dynamics, pages 3–22. Springer Netherlands, 2007.

[KAY+07] Youngmin Kim, Sanghyun Ahn, Hyun Yu, Jaehwoon Lee, and Yujin Lim. Proactive internet gateway discovery mechanisms for load-balanced internet connectivity in manet. In *Proceedings of the Information Networking. Towards Ubiquitous Networking and Services, International Conference (ICOIN 2007), Revised Selected Papers*, pages 285–294, Estoril, Portugal, 2007.

[KCC05] Stuart Kurkowski, Tracy Camp, and Michael Colagrosso. Manet simulation studies: The incredibles. *ACM SIGMOBILE Mobile Computing and Communications Review*, 9:50–61, 2005.

[KKR06] Ulrich Küster and Birgitta König-Ries. Discovery and mediation using diane service descriptions. In *Proceedings of the 3rd Workshop of the Semantic Web Service Challenge 2006 - Challenge on Automating Web Services Mediation, Choreography and Discovery*, Athens, GA, USA, November 2006.

[KKRM05] Michael Klein, Birgitta König-Ries, and Michael Müssig. What is needed for semantic service descriptions - a proposal for suitable language constructs. *International Journal on Web and Grid Services (IJWGS)*, 1(3/4):328–364, 2005.

[KKRO03] Michael Klein, Birgitta König-Ries, and Philipp Obreiter. Lanes – a lightweight overlay for service discovery in mobile ad hoc network. In *Proceedings of the 3rd Workshop on Applications and Services in Wireless Networks (ASWN2003). Berne, Swiss*, July 2003.

[Kle04] Michael Klein. Comparison of overlay mechanims for service trading in ad hoc networks. Technical report, University of Karlsruhe, 2004.

[KT03] Ulas C. Kozat and Leandros Tassiulas. Network layer support for service discovery in mobile ad hoc networks. In *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003)*, 2003.

[LHK05] Pasi Lassila, Esa Hyytiä, and Henri Koskinen. Connectivity properties ofrandomwaypoint mobility model for ad hoc networks. In *Proceedings of the 4th Annual Mediterranean Workshop on Ad Hoc Networks (Med-Hoc-Net 2005)*. Springer Science+Business Media, June 2005.

[LL04] Yawei Li and Zhiling Lan. A survey of load balancing in grid computing. In *Proceedings of the Computational and Information Science, First International Symposium(CIS 2004)*, pages 280–285, 2004.

[LNR04] Guolong Lin, Guevara Noubir, and Rajmohan Rajaraman. Mobility models for ad hoc network simulation. In *Proceedings of the 23rd Conference of the IEEE Communications Society (INFOCOM 04)*, 2004.

[LSN01] J. Lansford, A. Stephens, and R. Nevo. Wi-fi (802.11b) and bluetooth: Enabling coexistence. *Wireless Communications, IEEE*, 15(5):20–27, September/October 2001.

[LTEST08] Quan Le-Trung, Paal E. Engelstad, Tor Skeie, and Amirhosein Taherkordi. Load-balance of intra/inter-manet traffic over multiple internet gateways. In *Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia (MoMM 08)*, pages 50–57, New York, NY, USA, 2008. ACM.

[McH91] Roger McHaney. *Computer Simulation: A Practical Perspective.* Academic Press, Inc., first edition, 1991.

[MHKS08] Erling Matthiesen, Ossama Hamouda, Mohamed Kaaniche, and Hans-Peter Schwefel. Dependability evaluation of a replication service for mobile applications in dynamic ad-hoc networks. *Lecture Notes in Computer Science*, 5017/2008:171–186, 2008.

[MNR05] N. Mendelsohn, M. Nottingham, and H. Ruellan. Soap message transmission optimization mechanism. Technical report, W3C technical reports, http://www.w3.org/TR/, 2005.

[Mor02] Robert Morrow. *Bluetooth: Operation and Use*. McGraw-Hill, Inc., New York, NY, USA, 2002.

[MP02] Merritt Maxim and David Pollino. *Wireless Security*. Osborne/McGraw-Hill, Berkeley, CA, USA, 2002.

[MPV06] Vidal Martins, Esther Pacitti, and Patrik Valduriez. Survey on data replication in p2p systems. Technical report, INRIA, Rennes, France, December 2006.

[MS08] C. Mbarushimana and A. Shahrabi. Congestion avoidance routing protocol for qos-aware manets. In *Proceedings of the International Wireless Communications and Mobile Computing Conference (IWCMC 2008)*, Crete Island, Greece, 2008.

[MWV00] N. Malpani, J. L. Welch, and N. Vaidya. Leader election algorithms for mobile ad hoc network. In *Proceedings of the 4th international workshop on Discrete algorithms and methods for mobile computing and communications*, pages 96–103, 2000.

[New02] Eric Newcomer. *Understanding Web Services: XML, WSDL, SOAP, and UDDI*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.

[On04] Giwon On. *Quality of Availability for Widely Distributed and Replicated Content Stores*. PhD thesis, Technischen Universität Darmstadt, Juni 2004.

[OPN] OPNET. *OPNET Modeler Documentation Set, Version: 15.0 http://www.opnet.com*.

[OW09] Stephan Olariu and Michele C. Weigle. *Vehicular Networks: From Theory to Practice*. Chapman & Hall/CRC, 2009.

[PBRD03] C. Perkins, E. Belding-Royer, and S. Das. Rfc3561: Ad hoc on-demand distance vector (AODV) routing, http://www.ietf.org/rfc/rfc3561.txt, 2003.

[PGVA08] Prasanna Padmanabhan, Le Gruenwald, Anita Vallur, and Mohammed Atiquzzaman. A survey of data replication techniques for mobile ad hoc network databases. *The VLDB Journal*, 17(5):1143–1164, 2008.

[PS00] M. Papadopouli and H. Schulzrinne. Seven degrees of separation in mobile ad hoc networks. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM 00)*, volume 3, pages 1707–1711, 2000.

[PS01] Maria Papadopouli and Henning Schulzrinne. Effects of power conservation, wireless coverage and cooperation on data dissemination among mobile devices. In *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc 01)*, pages 117–127, New York, NY, USA, 2001. ACM.

[PWSK07] Guenter Prochart, Reinhold Weiss, Reiner Schmid, and Gerald Kaefer. Fuzzy-based support for service composition in mobile ad hoc networks. In *Proceedings of the IEEE International Conference on Pervasive Services (ICPS07)*, pages 379 –384, 2007.

[RDKK07] Amita Rani, Mayank Dave, Kurukshetra, and Kurukshetra. Performance evaluation of modified aodv for load balancing. *Journal of Computer Science*, 3:863–868, 11 2007.

[S. 99] S. Corson and V. Park. Internet-Draft: Temporally-Ordered Routing Algorithm (TORA) - Functional Specification. http://tools.ietf.org/html/draft-ietf-manet-tora-spec-02, 1999.

[SFBH09] Oussama Souihli, Mounir Frikha, and Mahmoud Ben Hamouda. Load-balancing in manet shortest-path routing protocols. *Elsevier: Ad Hoc Networks*, 7(2):431–442, 2009.

[SHHN06] Masako Shinohara, Hideki Hayahi, Takahiro Hara, and Shojiro Nisho. Replica allocation considering power consumption in mobile ad hoc networks. In *Proceedings of the 4th annual IEEE international conference on Pervasive Computing and Communications Workshops (PERCOMW06)*, page 463, Washington, DC, USA, 2006. IEEE Computer Society.

[SMZ07] Kazem Sohraby, Daniel Minoli, and Taieb Znati. *Wireless Sensor Networks: Technology, Protocols, and Applications.* Wiley-Interscience, 2007.

[SSB99] Raghupathy Sivakumar, Prasun Sinha, and Vaduvur Bharghavan. Cedar: A core-extraction distributed ad hoc routing algorithm. *IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS*, 17:1454–1465, 1999.

[SSG07] Min Song, Sachin Shetty, and Deepthi Gopalpet. Coexistence of ieee 802.11b and bluetooth: an integrated performance analysis. *Mob. Netw. Appl.*, 12(5):450–459, 2007.

[SZC+08] Weiwei Sun, Zhuoyao Zhang, Weiyu Chen, Bo Peng, and Yingxiao Xu. Decentralized execution of composite service in manets. In *Proceedings of the 2008 IEEE Asia-Pacific Services Computing Conference (APSCC '08)*, pages 355–360, Washington, DC, USA, 2008. IEEE Computer Society.

[TABR04] Vineet Thanedar, Kevin C. Almeroth, and Elizabeth M. Belding-Royer. A lightweight content replication scheme for mobile ad hoc environments. In *Proceedings of the Networking Technologies, Services, and Protocols (NETWORKING 2004)*, pages 125–136, 2004.

[TK09] Sasu Tarkoma and Jaakko Kangasharju. *Mobile middleware: architecture, patterns and practice.* WILEY, first edition, 2009.

[TMMF07] Peter Tröger, Harald Meyer, Ingo Melzer, and Marcus Flehmig. Dynamic provisioning and monitoring of stateful services. In *Proceedings of the 3rd International Conference on Web Information Systems and Technology (WEBIST 07)*, pages 434–438, Barcelona, Spain, 2007. Springer.

[TRF+07] Ioan Toma, Dumitru Roman, Dieter Fensel, Brahmanada Sapkota, and Juan Miguel Gomez. A multi-criteria service ranking approach based on non-functional properties rules evaluation. In *Proceedings of the 5th international conference on Service-Oriented Computing (ICSOC 07)*, pages 435–441, Berlin, Heidelberg, 2007. Springer-Verlag.

[VDI+03] Sudarshan Vasudevan, Brian Decleene, Neil Immerman, Jim Kurose, and Don Towsley. Leader election algorithms for wireless ad hoc networks. In *Proceedings of the DARPA Information Survivability Conference and Exposition*, pages 22–24, 2003.

[VP05] Christopher N. Ververidis and George C. Polyzos. Routing layer support for service discovery in mobile ad hoc networks. In *Proceedings of the 3rd IEEE International Conference on Pervasive Computing and*

Communications Workshops (PERCOMW 05), pages 258–262, Washington, DC, USA, 2005. IEEE Computer Society.

[VP08] Christopher N. Ververidis and George C. Polyzos. Service discovery for mobile ad hoc networks: A survey of issues and techniques. *IEEE Communications Surveys and Tutorials*, 10(1-4):30–45, 2008.

[VRdL05] A. Varshavsky, B. Reid, and E. de Lara. A cross-layer approach to service discovery and selection in manets. In *Proceedings of the IEEE International Conference on Mobile Adhoc and Sensor Systems Conference*, pages 8 pp. –466, 7-7 2005.

[WC02] Brad Williams and Tracy Camp. Comparison of broadcasting techniques for mobile ad hoc networks. In *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc 02)*, pages 194–205, New York, NY, USA, 2002. ACM.

[WL02] K. Wang and B. Li. Efficient and guaranteed service coverage in partitionable mobile ad-hoc networks. In *Proceedings of the IEEE Computer and Communications Societies 21st Annual Joint Conference (INFOCOM 2002)*, New York, USA, 2002.

[WPS+00] M. Wiesmann, F. Pedone, A. Schiper, B. Kemme, and G. Alonso. Understanding replication in databases and distributed systems. In *Proceedings of the 20th International Conference on Distributed Computing Systems (ICDCS 2000)*, page 464, Taipei, Taiwan, 2000.

[YMH05] H. Yu, P. Martin, and H.S. Hassanein. Cluster-based replication for large-scale mobile ad-hoc networks. In *Proceedings of the International Conference on Wireless Communications and Mobile Computing*, 2005.

[ZSL04] Jing Zheng, Jinshu Su, and Xicheng Lu. A clustering-based data replication algorithm in mobile ad hoc networks for improving data availability. In *Proceedings of the 2nd International Symposium on Parallel and Distributed Processing and Applications (ISPA 2004)*, pages 399–409, 2004.

Glossary

Average velocity The average velocity of the moving nodes influenced by the Random Waypoint mobility model. 147, 150, 151, 234

Composite response time The average round trip time for the succeeded composite requests between posing the composite service request and receiving the complete response of this composite request. 213

Composite success ratio The ratio of the correctly served and succeeded composite service requests to the total number of the generated composite service requests regarding a specified length of a composite service request. 213, 214

Connected session Is a virtual connection between a client and a server which may have different statuses and attributes based on the client request parameters regarding a specified service. 63, 66, 109, 186–189, 191, 193, 196, 197, 234

Correctness ratio Is an measurement for how close a service distribution is to an optimum assumed distribution. 90–92, 98, 114–116, 120, 124, 125, 129, 167, 193, 196, 198, 234

Linear Correctness Ratio (LCR): A correctness ratio based on a linear relation between the number of active replicas inside a given network to the number of its partitions at a given time. 91, 114, 233

Network density A relation between the number of network participants, their transmission ranges, and the area that this network is deployed in. 86, 126, 198, 225

Prevalence Is the ratio of the size of the active replica set to the network size. 70–72, 80, 84–86, 88, 89, 92–95, 97, 116, 132, 136, 140–142, 153, 158, 159, 168, 193, 195–199, 201, 218, 225–227

Rational Correctness Ratio (RCR): A correctness ratio based on a rational relation between the number of active replicas inside a given network and the number of its partitions at a given time. 91, 114

Replication cost The number of the actively deployed servers in the network at a certain time. 64, 93, 110, 225

Replication degree The replication degree is the number of replicas required for a data item or any resource to ensure a certain availability in the network. 92, 93, 116, 120, 226

Requirements' Index Is an attribute that combines the service popularity to the volume of requirements (of resources) required to enable a client to host a replica of a specified service. 67, 110, 153, 176, 192, 235

Residence time The average time that a service or a replica stayed running on a hosting node. 70, 72, 84, 89, 116, 136, 140, 153, 158, 159, 226

Service response time The time interval required for a server to produce a service response regarding a specific request. 193, 211

Success ratio Is the ratio of the number of successful service requests to the overall number of requests generated in the entire network. 84, 87, 88, 116, 132, 136, 141, 153, 158, 159, 161, 166, 167, 175, 178, 179, 181, 193, 195, 197–199, 201, 211

Transmission range In terms or length, it is the radius of a circular coverage area that enables the processes of packet transmitting and receiving between two wireless nodes. 86, 94, 115, 147, 162, 176, 212, 218

Typical Partition The most weighted network partition in terms of the partition sizes in a given network at a given time. 114, 126, 129, 136, 140–142, 157, 158, 198, 226

Weighted Linear Correctness Ratio (WLCR): A correctness ratio based on a linear relation between the number of active replicas inside a given network, the weight of the network partitions compared to the network size, the number these partitions at a given time. 114

Weighted Rational Correctness Ratio (WRCR): A correctness ratio based on a rational relation between the number of active replicas inside a given network, weight of the network partitions compared to the network size, the number these partitions at a given time. 115, 189, 233

Weighted Rational Correctness Ratio for Load Distribution ($WRCR_L$): A weighted rational correctness ratio that ignores the activated replicas to intake the excess of requests regarding the originally activated servers in computations of the service distribution optimality. 189

Appendix

APPENDIX A

SDP Primary Implementation in Microsoft® VisualC++®

SDP Primary Implementation in Microsoft® VisualC++®[1]

The Service Distribution Protocol for MANETs

FRIEDRICH-SCHILLER-UNIVERSITY OF JENA

2008-2009

[1] Version 8.0.50727.42 (RTM.050727-4200)

Table of Contents

Contents

Introduction

Please refer to [1] to have more details about the basic concepts and ideas of the **S**ervice **D**istribution **P**rotocol (SDP). It was required to build a simulation tool that enable the founders of SDP to estimate their early proposals about service replication in MANETs. Avoiding the complications of building a set of network layers, components, and architectures was highly required. Since, one of the most important SDP advantages is being an architecture independent protocol, it assumes that -for example- the presence of any specified network architectures or components are not a big deal. So, at the beginning of the research effort of estimating the feasibility of SDP, two SDP simulation tools have been introduced. These tools have been built in Microsoft® VisualC++®.

The proposed SDP simulations tools are supposed to give simulations for the performance in the best operation cases. In this tool, a suitable and efficient routing, transportation and scheduling network components have been assumed

The two proposed simulation tools are oriented to answer more the questions of when and where to allocate replicas and how these replicas are allowed to continue active than to answer the questions of how to perform the replication itself. So, both assume the presence of a routine that takes care about the required operations for the service synchronization. Both of the delivered SDP simulation tools cannot model the behavior of the deployed services. So, they assume that the response will be sent to the requester clients once the request reaches the provider side.

The deployed services by both of these simulation tools are replicable, they are not resource-competing at the provider sides. The concepts of managing the excess of the service requests for the providers and balancing the load between the concurrent active replicas are not addressed in both. More sophisticated simulation implementations are introduced in SDP-simulations in Opnet® Modeler® in [2] and [3].

The terms of “request” and “call” are exchangably used in this documentation. Also, “requesting” and “calling” behavior are exchangably used.

Contents:

This documentation consists of 2 parts: I. Release Notes and II. User Manual In the first part, a general descriptions about the requirements and abilities of the two delivered SDP simulation tools are given. Part II shows a user manual which introduces how to use the delivered tools and how to interpret the results.

What to download? How to Run?

What to download?

A compressed ".rar" file is published on "http://hnsp.inf-bb.uni-jena.de/professur" contains two Microsoft®VisualC++® projects. The two project files contain the source code of both implementations.

The first project "Modes" is oriented to estimate the performance of SDP for only one deployed service where the mobile nodes apply the Random Waypoint Mobility model [4]. In this implementation, the whole set of service election and selection modes of [5] are modeled. The second project "Graph" introduces a simulation of more realistic mobility.

How to install?

Download a proper version of Microsoft® VisualC++® from "http://www.microsoft.com/express/Downloads/". Install the downloaded version of Microsoft® VisualC++®. Open the Microsoft® VisualC++® . Open any of the project files and accept any recommended conversion for the compatibility of the project. Now, you are ready to run a simulation.

Part I: Release Notes:

The requirements, abilities and limitations of the delivered SDP simulation tools are given in this part.

It requires:

- Software:
  - Microsoft® Windows XP® or any compatible version.
  - Microsoft® VisualC++® compiler with Visual studio® suite 2005 or higher.

- Hardware:
  - Suggested machines by:
    - Microsoft®, for the operating system and Microsoft® Visual Studio®.

Abilities:

The provided SDP simulation tools can simulate in the best case of network operation the performance of SDP for only one deployed service and its resultant replicas during the service distribution processes.

The first SDP simulation tool which is a “Random Waypoint” based simulation can visualize the network participants and how the network partitioning behavior. This feature is disabled in the second project. Features of the service selection modes in [6] are not implemented in the second project file “Graph”.

Limitations:

The following features are missingin the deliverable implementations:

- Simulation of multiple deployed services in the same network
- Modeling of the service behavior (like the response time [7])
- Modeling of the request-response processing (like the connected sessions [7])
- Modeling of data transportation and routing and their effects of SDP performance
- Modeling of the resources on both of the network and participants [7]

Part II: User Manual:

This user manual aims to enable any interested person to estimate the performance of a typical SDP-based service. SDP performance is considered to be in terms of the performance metrics which are given in [8]. A step by step guidance for how to run a SDP simulation and get the results to be analyzed is presented in this section. Moreover, a brief description about the structure of the two delivered simulation tools is introduced.

Main Entry? How to Compose Experiments?

Both of the project files supply Microsoft® Windows® applications for acquiring the specifications of the primary SDP based simulations. As in Figure 1, the shown form is the main entry form for acquiring and managing the proposed SDP simulations. On the upper left part of this form, the "Data collection option" group is located. This option determines the independent parameter in experiments of a simulation and enables the related group of specifications. Each of these groups is enabled mutually when, the related option of "Variant size", "Variant mobility", "Variant prevalence" or "Correctness Analysis" is selected from the "Data collection option" group. Enabling the option of "Variant prevalence" implies enabling both groups of "Prevalence settings" and "Auto PPP".

Network size setting

The variant size option enables the little bit lower "Network size settings" group where, the user is enabled to draw the limits of the starting network size (number of the network participants), the maximum allowed network size and the step of the increment in nodes. When the "variant size" option is enabled, the "Mobility settings", "Prevalence settings", "Auto PPP" and "Correctness Ratio Analysis" groups are disabled and only the "Network size settings" group is enabled. By settings the shown specifications in the "Network size settings" group in Figure 1, the simulator will perform the whole simulation for a network size of 10 mobile nodes then 20, 30, … and 140 nodes. The number of runs time inside one simulation step is fixed to 20 runs.

Mobility Settings

The "Mobility settings" group enables the user to supply the specifications for the Random Waypoint mobility model which is used in the SDP primary simulation for in the project file "Modes". The other project file of "Graph" applies another mobility model and will be discussed later. The speeds are selected randomly and uniformly between 0 m/s and the value of the "Max. Speed" field. Also, the same for the pause-moving intervals which are distributed randomly and uniformly between 0 (minutes) the value of the "Max. Pause Time" field. The mobility index tunes the given speed and pause specifications. For example, if the mobility index is 60% then the speeds will be selected between the interval of 0 m/s and (Max.Speed x mobility index / 100) and the pause time intervals will be between 0 minutes and (Max.PauseTime x (1 – mobility index / 100)). So, for the given specifications in Figure 1, for the "Mobility settings" group (and if the "Variant mobility" option is selected), the start MAX.speed will be 6 m/s and the start Max.PauseTime will be 15 minutes for the first experiment in the simulation. Then, during each next step the mobility index will be increased by 10%.

The size of the network will be fixed during all of the experiments. It will be always the value of the "start network size" field in the deactivated "Network size setting" group.

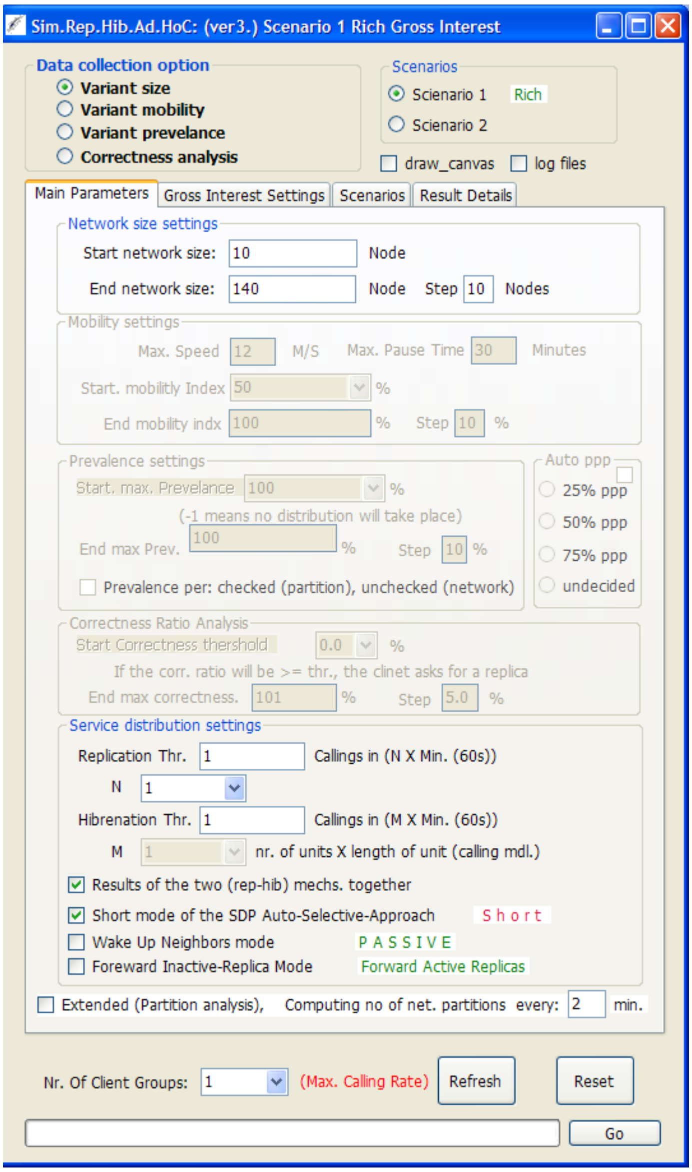


Figure 1: The main form of a SDP simulation.

Prevalence Settings

The definition of service prevalence is described in details in [8]. The service prevalence is an indication for the service cost [9] but on the level of the whole network. It indicates the ratio between the number of the mobile nodes who have an active service to the total number of the network participants (network size). The delivered SDP primary simulation can perform a service prevalence analysis as it is going to be presented. The defaulted allowed maximum prevalence in the network is 100%. This allowed maximum prevalence ratio can be tuned if the “Variant prevalence” option is selected from the “Data collection option”. In this case, the network size is fixed to the value of the “start network size” field in the deactivated “Network size setting” group. Also, the same for the mobility settings, speeds , pause intervals and the mobility index will be fixed to the current values of their relative fields in the “Mobility settings” group.

The prevalence is by default restricted on the level on network as previously defined. The lower check box of “Prevalence” can toggle the prevalence level to be restricted to the level of network partition. Beside the group of “Prevalence settings”, the group of “Auto PPP” is allowed actively where the user can specify pre-chosen values for the “PPP” Prevalence Per Partition. This option may be activated from the check box of “Auto PPP”.

Correctness Ratio Analysis

Applied based on the defined LCR (Linear Correctness Ratio [8]), can be used for restricting a specified allocation correctness ratio based on any drawn correctness ratio computation method [8].

Service Distribution settings

In Figure 1, the main parameters of SDP [1,8] replication and hibernation threshold are specified in the group of “Service distribution settings”. The field of “Replication Thr.” Indicates the number of the calls or requests to be posed from a mobile node during the value of “N” field as a multiplier of the minutes. The same thing for the hibernation threshold which optioned from the values of “Hibernation Thr.” and “M” fields.

The check boxes in the “Service distribution settings” group can be explained as follows:

- “Results of the two(rep-hib) mech. together”: if checked, the simulator applies both of the replication and hibernation together in the experiments otherwise, only the replication mechanism will be applied.
- “Short mode ….”: if checked, the short service election mode will be applied during the hibernation evaluation test of by the service providers otherwise, the long election mode will be applied [5].
- “Wake Up ….”: if checked, a “WakeUp” mode will be activated for selection of the nearest hibernated service provider [6].
- “Forward Inactive …”: If checked, the forwarded replicas will be not actively allocated by their hosting mobile nodes. They will be only cached.

Topology Analysis

During performing the experiments, the simulator can observe and save some information about the current network topology like: the number of network partitions. This option is activated for all options of the “Data collection option” group through the check box of “Extended (Partition analysis)”. The simulator will probe the network for obtaining the required topology data each time interval. This time interval is specified through the value of the “Computing no of net. Partition every:” field in minutes.

How to specify?

Simulation Time

The simulation time is recommended to be set relatively to the values of the requesting (calling) model (behavior). As in Figure 2, the user is allowed to set the simulation time (in minimum resolution of seconds) through the “Simulation Time” group.

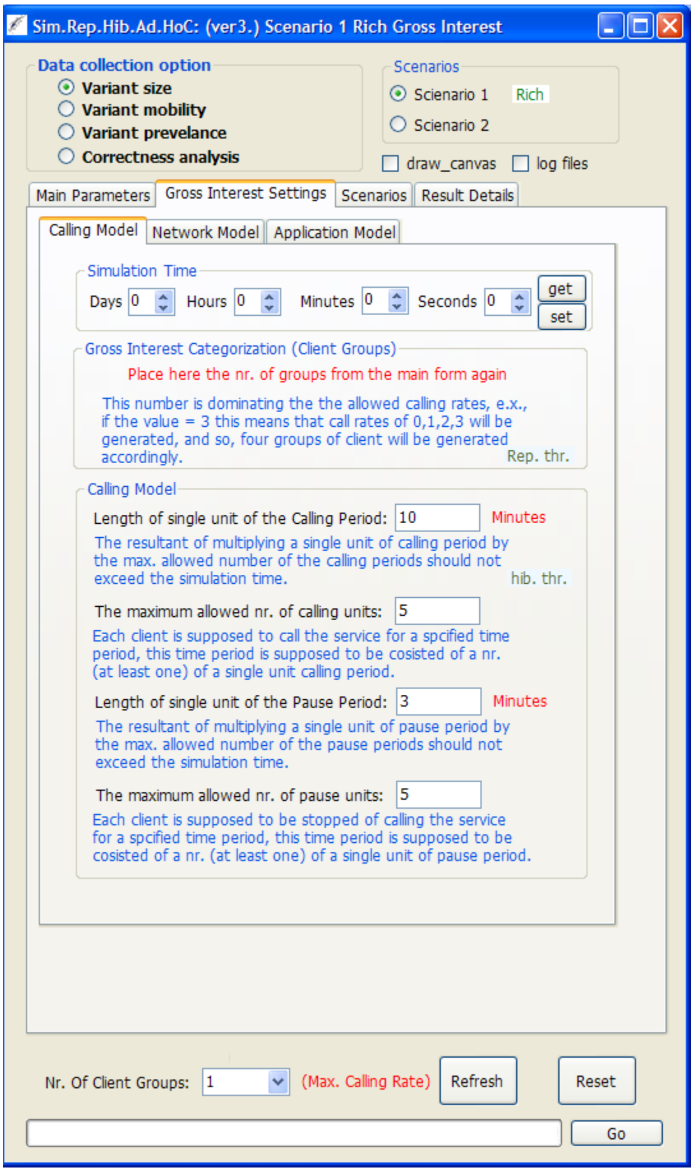


Figure 2: Specifying the service interest and requesting behavior

Service Interest Scenarios

In Figure 2, in the upper right part of the background of the main form, in the "Scenarios" group, two main pre-set calling behavior settings based on the calling model of [8] are given. The first scenario id the "Rich" one where the client nodes generate a high number of frequent service requests regarding the providers while, in the second "Poor" scenario, the client are supposed to generate a rare number of infrequent service requests. By, toggling the selection of the radio buttons, the values of the "Calling Model" tab changes accordingly.

On the main form, the combo box "Nr. Of Client Groups" determines the requesting rate and accordingly the number of client groups. More details about the meaning of these values and their effects can be found in [8].

Network Area Features and Nodes' Transmission Ranges

The SDP primary simulator tool of the project file "Modes" considers that the network is deployed in a rectangular flat area with no obstacles. By selecting the "Gross Interest Settings" tab and choosing the "Network Model" sub-tab. As in Figure 3, the user is allowed to specify both of each node transmission are radius and the network area dimensions in meters.

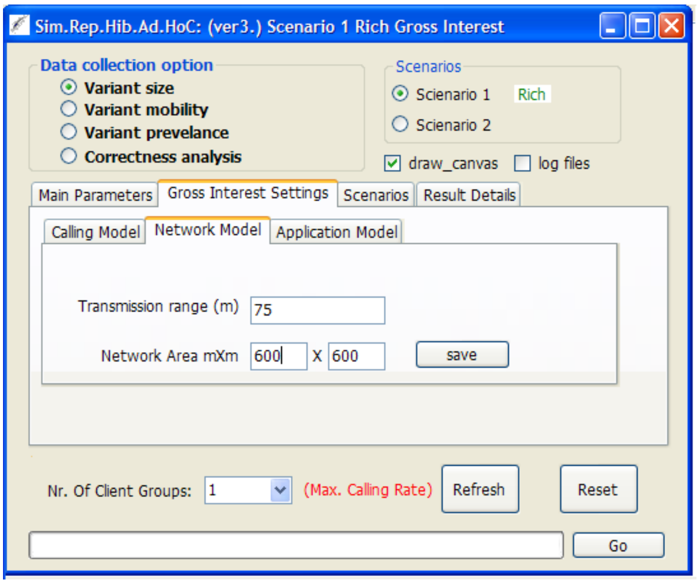


Figure 3: Specifying the network are and the node transmission ranges

Other Tabs

In Figures 1, 2 and 3, the other main tabs of "Scenarios" and "Results Details" and the Sub-tab "Application Model" are reserved for further future functionalities.

Tracing The Results

Both of the provided SDP simulation tools provide the results in plain text files. It provides all of the performance metrics stated in [8].

Structure of the Output Files

Each of the results' measurements (performance metrics) is presented in some specified named output file as a field. As in Figure 4, each of the output files consist a set of fields which are separated by a tab ('\t' character in C++). Each of these fields is divided into six columns. The abbreviations of "Avg", "StD", "Max." and "Min." mean "the average value", "the observed standard deviation", "the maximum observed reading" and "the minimum observed reading" of a measurement respectively. These columns are separated by a tab ('\t' character in C++).

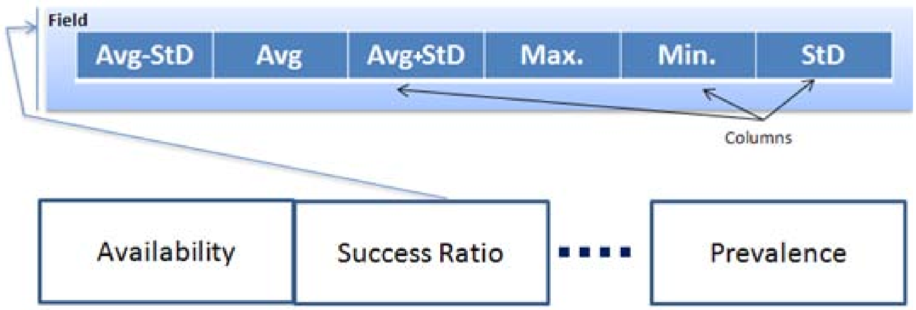


Figure 4: Structures of the fields in the output files

Interpreting the Output Files

Each of the output files has a fixed name and contains a fixed set of the performance metrics. Changing the names of these files are only possible by traversing the source code to the header file of "counters.h" and its implementation in "counters.cpp".

Mainly there are three main files which contain the following measurement groups:

1- file name "**repl_overlaps_abst_XLS.txt**" holds the following fields:
<Success Ratio> <Service Availability> <Service Residence> <Service Prevalence>

2- file name "**variant_allocation_correctness_xls.txt**" holds the following fields:
<LCR> <WLCR> <RCR> <WRCR> <Typical Partition>

3- file name "**part_analysis_xls.txt**" holds the following fields:
<Number of Partitions> <LCR> <Replication Degree>

The previously mentioned files are automatically written and saved and should be found in the same folder of the executable file.

Run

Now, you are ready to design experiments, specify your SDP primary simulation and interpret the simulation results. Just press on "Go" and the drawn indicator will show the progress of the individual runs on the main form.

How to Run a Visual Demo?

The provided SDP simulation tool in the project file "Modes" can present a visualized simulation for the process of service distribution based on the provided user specifications . On the main form, Figure 1, a user can activate the "draw_canvas" check box. Once the simulation starts, a canvas form will pop up showing the modeled network as in Figure 5. The network partitions(dashed lines-clusters), mobile nodes(transparent rectangles), interested clients (green rectangles), providers (red rectangles) and nodes with hibernated replicas (smaller black rectangles). On the right hand part of this canvas form, a brief about network specifications and current achieved results is shown. By clicking on any of the mobile nodes at any time, the simulation will halt and show a message box with the basic information of this node.

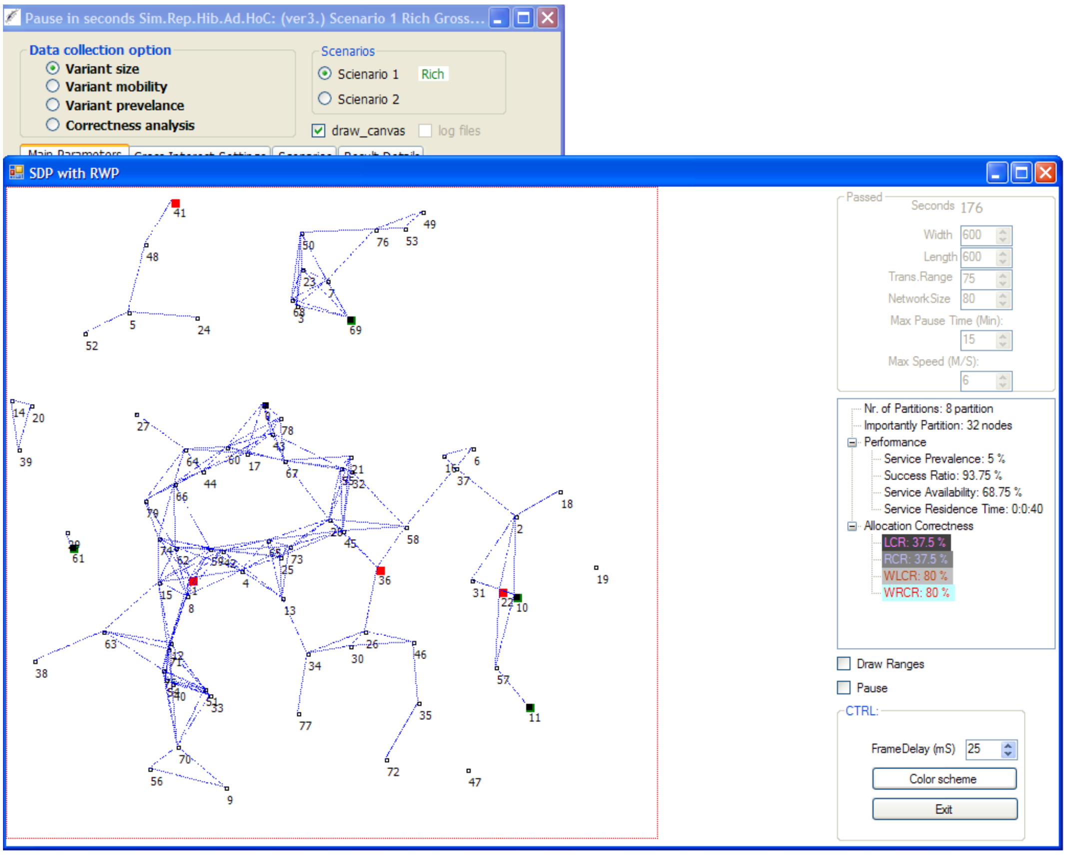


Figure 5: Running of a visualized simulation for SDP

By pressing "Exit", in Figure 5, the shown canvas form will be closed and the simulation will continue performing the specified experiments from the main form (Figure 1).

How to Run a Graph Based Area Mobility Model Simulation?

The Graph Based Area mobility model is more realistic mobility model [10, 11]. The provided SDP simulation tool of the delivered project file of “Graph” can model this mobility model and capture the performance metrics in this case. Keep in mind that, some of the previously mentioned features are disable in this project file. No visualized simulations can be performed here. Also, the extended service selection modes of [6] are not included here. In the main form, a check box of “Graph_based Mobility” is presented. By checking this box, another form will pop up where the user is enabled to specify the places , the places’ topology , the mobility inside these places.

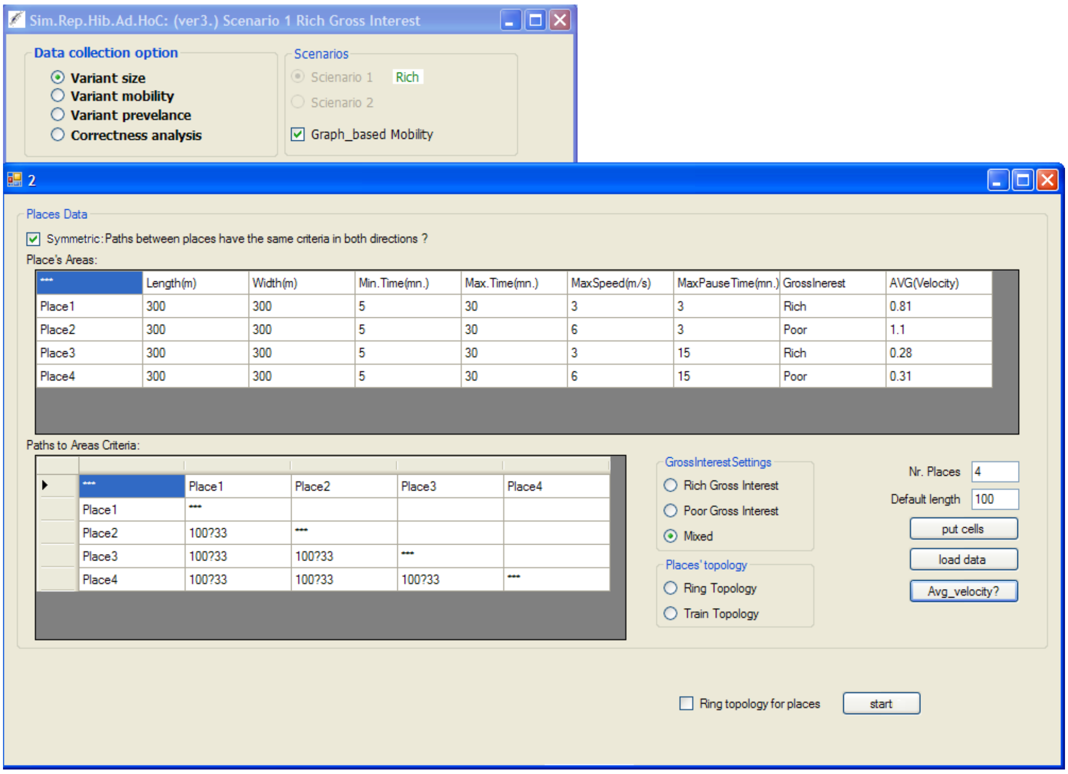


Figure 6: Graph area based mobility model specification for SDP

Specifying Places and Corridors

In the Graph Based Area mobility, the network is placed in a set of places. The places are connected through a set of corridors. A corridor allows bidirectional movements for the mobile nodes between two places. As shown in Figure 7, the formation of wireless links between the mobile nodes is only allowed for those nodes which are in the same place or the same corridor. The “Places Data” group, showed in Figure 6, enables the user to specify features of the places (assume rectangular shaped places). The random waypoint mobility model are assumed to manage the mobile nodes’ movements in each place with different considerations for the mobility specifications (like speed and pause time) and the requesting behavior. The number of places is set by specifying the value of the “Nr. Places” field and by pressing "put cells". Once, the data are supplied for each of the places, the average velocity [11] of each place can be computed by pressing on "Avg_velocity?".

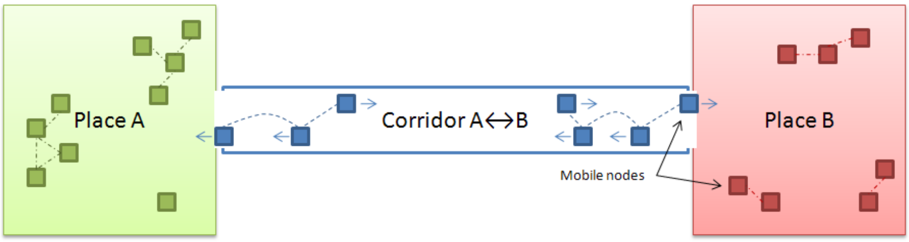


Figure 7: Mobile nodes move between two places in a graph-based area mobility model

The Places' Topology

Based on the check box of "Symmetric: …" in the "Places Data" group, the simulator gives the same acquired features of the user specifications for the corridors between the places in both directions.

By clicking on each of the cells appearing in the "Paths to Area Criteria" data grid, a guiding footer which acquires the length of the corridors in meters and the probability (weight) of being selected by a node inside a place (which is connected to another place by this corridor) to traverse to another place once its "time of stay" expires in the current place.

The "Places' topology" group can specify a set of per-set topologies for the places. The user can using the data grid in "Paths to Area Criteria" set whatever desired topologies.

Programmer Guidance

We recommend that, if you have any proposals to be applied with SDP, try to apply your modifications in [2]or [3] to use the more detailed specifications there for more realistic MANET environments. But, if you require more details or explanation for the SDP-primary implementation source code, these information will be supplied upon request .

Nevertheless, we provide here a brief and abstract description about the provided source code structure as an introduction for a programmer guide. Only, the most important structures have been included in this section.

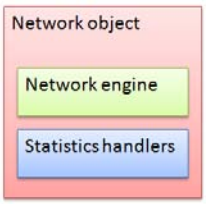


Figure 8: The abstract structure of a network object

Main Simulation Structures

Figure 8 and 9 show the main structures involved in the delivered SDP simulation tools. In Figure 8, the network is modeled as an object. Only one network instance is created per the simulation run. The network object includes a network engine which is responsible for triggering the mobile nodes to perform the required processes of SDP. Moreover, the set of main statistics handlers that enable probing and saving the performance matrices and other measurements is maintained in this object.

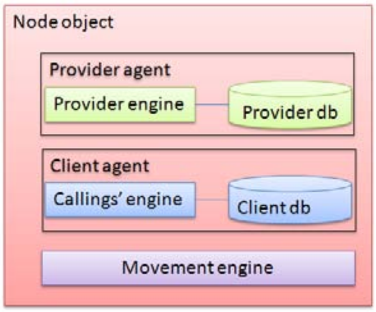


Figure 9: The abstract structure of a node object

In Figure 9, two operating agents are included by each node objects. The “Provider agent” is responsible for starting the provider engine upon request of the network object (engine). The provider engine is maintaining the required information about the offered services/replicas in a provider database. The information of the gained requests from the interested client to the active replicas are stored in the provider database.

The “Client agent” performs the required processes of generating request regarding the most interesting reachable service/replica. Based on an intial-defualt information about the service, from the beginning of the simulation, it starts evaluating the requests even if the service is not reachable. In this case, the requests are considered failed.

The “Movement engine” is supposed to update the location of the mobile node based on the specified mobility model in the simulation periodically. Based on the new computed locations, the network partitions are computed and the assumed routes between the providers and client are realized accordingly.

Simulation Flow

Figure 10 presents a general flow of the primary provided SDP simulation tools. As mentioned before, the number of runs for each simulation situation is fixed to 20 runs (“RequiredRuns” in Figure 10) as a default value. This value can be changed through the deliverable projects’ source code.

The simulation creates a network N and the required network size (nodes) at the initial time. Each time step, the network engine requires each node to update its location information based on its mobility specifications then, to trigger the client agent to update its client database (about where are the most interesting reachable service/replica located) then performs the request generation process if any. The required replication tests are done here.

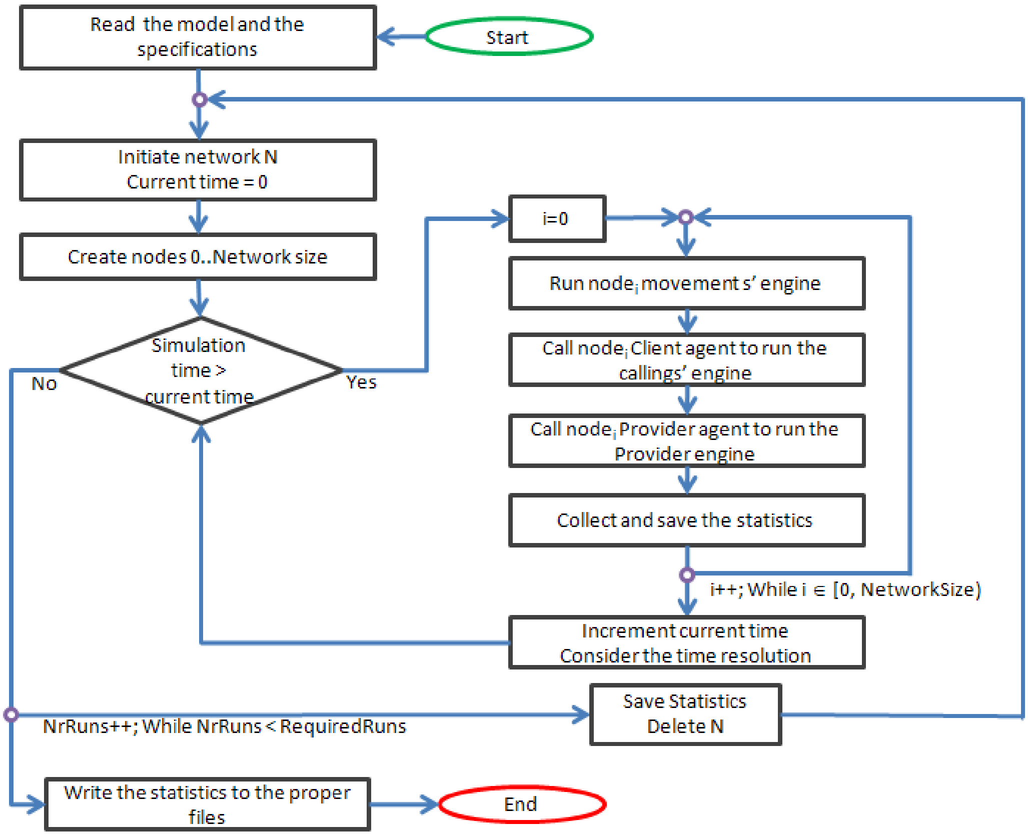


Figure 10 : A simulation general flow of the SDP primary implementations

Then, the node triggers its provider agent which performs the required hibernation tests based on the available information about the gained requests to the offered service/replica. The provider agent is responsible also for processing the received requests about hosting replicas by the interesting clients.

After satisfying all the generated events by both of clients and providers and collecting and updating the performance metrics, the general simulation current time should be incremented and the network engine is called again and so on. Once the simulation current time is greater than the required time ("Simulation time" as in Figure 10), The current network performance metrics and measurements are saved and then the whole resources allocated to this network instance N are deleted and the number of runs ("NrRuns" in Figure 10) is incremented. While, the number of runs is less than a fixed required number of runs, a new network with a new set of nodes are contained in the previously mentioned simulation processes and so on.

At the end of the simulation processes, the performance metrics should be written in their destination files as mentioned before.

References

[1] Mohamed Hamdy and Birgitta König-Ries, Book of Communications in Computer and Information Science, Book of the selected papers of the ICETE 2008. The Service Distribution Protocol for MANETs- Criteria and Performance Analysis, Springer Berlin Heidelberg, 2009, 48, 467-479.

[2] Mohamed Hamdy and Birgitta König-Ries. SDP-Implementation in Opnet® v.1: MANETs' Service Replication and Load Balancing -Release Notes-User Manual–Programmer Guide, https://enterprise1.opnet.com/tsts/4dcgi/MODELS_FullDescription?ModelID=944, FRIEDRICH-SCHILLER-UNIVERSITY OF JENA, 2010.

[3] Mohamed Hamdy and Birgitta König-Ries. SDP-Implementation in Opnet® v.2: SDP-Based Composite Service Execution in MANETs -Release Notes-User Manual–Programmer Guide, https://enterprise1.opnet.com/tsts/4dcgi/MODELS_FullDescription?ModelID=951, FRIEDRICH-SCHILLER-UNIVERSITY OF JENA, 2010.

[4] Guolong Lin et al, "R. Mobility Models for Ad hoc Network Simulation," The 23rd Conference of the IEEE Communications Society (INFOCOM 2004), Hong Kong, China.

[5] Mohamed Hamdy and Birgitta König-Ries, "Leader Election Modes of the Service Distribution Protocol for Mobile Ad Hoc Networks," The fourth German Community Conference on Mobility and Mobile Information Systems (MMS 2009), Münster, Germany.

[6] Mohamed Hamdy and Birgitta König-Ries, "New Service Selection Strategies for the Service Distribution Protocol for MANETs," The IADIS International Conference Wireless Applications and Computing (WAC 2010), Freiburg, Germany.

[7] Mohamed Hamdy and Birgitta König-Ries, "Interest-based Load Balancing Mechanism for MANETs," submitted/under reviewing.

[8] Mohamed Hamdy and Birgitta König-Ries, "Service Availability, Success Ratio, Prevalence, Replica Allocation Correctness, Replication Degree, and Effects of Different Replication/Hibernation Behavior Effects of the Service Distribution Protocol for Mobile Ad Hoc Networks -A Detailed Study-," JENAER SCHRIFTEN ZUR MATHEMATIK UND INFORMATIK, Technical Report: Math/Inf/08/08, Friedrich-Schiller-University Jena, December 2008.

[9] Padmanabhan et al, "A survey of data replication techniques for mobile ad hoc network databases", *The VLDB Journal, Springer-Verlag New York, Inc.,* 2008, 17, 1143-1164

[10] S. Bittner et al, "The area graph-based mobility model and its impact on data dissemination," The 3rd International Conference on Pervasive Computing and Communications (PERCOMW 05).

[11] Mohamed Hamdy and Birgitta König-Ries, "Impact of Heterogeneous Mobility Models on the Service Distribution Protocol for MANETs," The IADIS International Conference Wireless Applications and Computing (WAC 2010), Freiburg, Germany.

APPENDIX B

SDP-Implementation in Opnet® v.1: MANETs' Service Replication and Load Balancing(Release Notes - User Manual - Programmer Guide)

SDP-Implementation in Opnet® v.1

MANETs' Service Replication and Load Balancing

- Release Notes -User Manual –Programmer Guide

FRIEDRICH-SCHILLER-UNIVERSITY OF JENA

Acknowledgements

Many thanks are due to Alexander Klein from University of Wurzburg for his early support.

Table of Contents

SDP-Implementation in Opnet® v.1

- Release Notes -User Manual –Programmer Guide

Introduction

Enabling service oriented application on MANETs represents a common motivation for many research efforts during the last few years. MANET introduces many challenges. The main challenge for a service application there is the availability. There is no guarantee that any of the network participants will be able to access the service at any time of the network operation time. SDP (the **S**ervice **D**istribution **P**rotocol for MANETS) [1] has been introduced as an approach with many innovative mechanisms to increase the mobile service availability. It is mainly based on a new technique in servicer replication called "replication-hibernation-restoration". SDP can achieve a very efficient service distribution of the generated replicas (Service copies) over the MANET. In this report, the implementation of SDP in the Opnet® simulator is being described. More details about how SDP is functioning, its performance and comparisons can be found in [2].

Contents:

This documentation consists of 3 parts: I. Release Notes, II. User Manual and III. Programmer guide. In the first part a short description about the delivered software features is presented. Part II shows an easy user manual which introduces how to use the delivered software under the case tool of Opnet® Modeler® with Wireless. The last part show a detailed programmer guide to make it easy to implement your special extensions for the SDP-implementation in Opnet®.

Who is supposed to read this documentation? which part?

Mainly, apart from the release notes part, two categories of users may read this documentation. The purpose of reading such a documentation is varying between evaluating the SDP protocol under different specifications and settings and modifying SDP and its mechanisms. We guess that the first category, which we name "user category", is associated to the first purpose. If you are a user, then it is better, if you have some knowledge about:

- Opnet® Modeler® Wireless.
- Simulation Concepts.
- MANETs:
  - Routing protocols.
  - Data Transpiration protocols.
  - MANET addressing protocols.
- SOA Applications.

On the other hand, we call the second category the "programmer category" which is associated to the second purpose. Besides the knowledge of the required mentioned points in the previous category, the programmer should have knowledge about the following:

- C++.
- Automata.
- Modeling concepts in Opnet® Modeler®.
- Network architectures.
- Queuing techniques.

The first user category will easily read and understand the first two parts of this documentation. While the programmer category will be able to understand the whole documentation.

What to download? How to Install?

What to download?

Please find and download the SDP compressed model file in the Opnet contributed models at "https://enterprise1.opnet.com/tsts/support/cont_models.shtml".

The model compressed file is contacting the following components:

- A channel matching pipeline:
  - pipeline stage: dlvr chanmatch75m.ps.c
- The mobile node:
  - Process model: dlvr SerExe SDP Application Process Model.pr.m
  - Node architecture: dlvr ServExe SDP node arch.nd.m
- The repository node
  - Process model:dlvr SerEXE SDP global_db process model.pr.m
  - Node:dlvr SerEXE SDP global_db.nd.m
- A sample project:
  - Project1 SerEXE SDP 50 node.project.rar

How to install?

- Install Microsoft® Visual Studio® with VisualC++® 2005 or higher.
- Get a suitable Opnet® Modeler® license and install Opnet® Modeler® Wireless v.15 is preferred.
- Associate the C++ compiler to Opnet® Modeler®. Please have a look on www.opnet.com/support, if you have a question.
- Download the previously mentioned files in the "What to download?" section.
- Remove the "dlvr " token part from the naming of the downloaded files.
- Make a specified folder "SDP-Home" to host the downloaded files.
- Add the "SDP-Home" folder to the Opnet® Modeler® model directory by:

- From the project menu-bar, select: File→Manage model File→Add model directory.
- From the project menu-bar, select: File→Manage model File→Refresh model directories.

Start evaluating the SDP-implementation in Opnet®.

Part I: Release Notes:

In this part a short description about the requirements, abilities and limitations of the delivered software (SDP-Implementation in Opnet®) is given. This version is our initial release version, so the delivered release notes do not refer to any previous versions.

It requires:

- Software:
  - Opnet Modeler® with Wireless version 15 and its default delivered model libraries.
  - Microsoft® Windows XP® or any compatible version.
  - Microsoft® VisualC++® compiler with Visual studio® suite 2005 or higher.

- Hardware:
  - Suggested machines by:
    - Opnet®, for Opnet® Modeler®.
    - Microsoft®, for the operating system and Microsoft® Visual Studio®.

Abilities:

The provided SDP-Implementation in Opnet® present an Opnet® module which can perform all the functionalities of service distribution and the related mechanism with a user friendly Opnet® interface.

It can:

- Deploy either one or different services in a MANET
- Model a variant Gross Interest [2] for a specified service or set of services based on a very fixable modeling approach .
- Associate the client Gross Interests to the offered services.
- Distribute a set of initial replicas (the original service and a set of initial copies) of each service on the mobile nodes automatically.
- An easy and clear SDP-modes and specifications acquisition.
- Perform the service replication-hibernation-restoration mechanisms.
- Expressive collecting of different statistics for measuring the performance of SDP such as: Service availability, SDP availability, execution success ratio, replica allocation correctness, replication cost, typical network partition, …etc.

Limitations:

- Regarding Opnet® Modeler with Wireless® version15 release notes, the delivered software inherits the limitations described there. For example, when deploying a set of mobile nodes in a MANET (apart from their types), Opnet® can only operate only one routing protocol for the network.
- The default version of SDP-Implementation in Opnet® supports at most 10 running simultaneous different original initial services (apart from their replicas or initial sites). If you have a simulation specification which needs more number of initial services, please contact us.

- Network operation time is supposed not to exceed 24 hours (one day). If you have a longer-time scenario, please contact us to supply a solution for that.

Part II: User Manual:

Basic Concepts

SOA

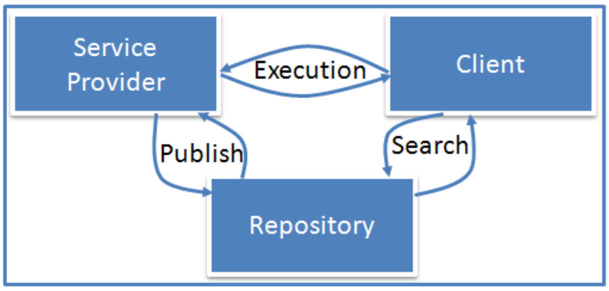


Figure 1: A general SOA architecture

SDP assumes that the services are organized and handled under the standards of SOA (**S**ervice **O**riented **A**rchitecture). Many proposals for achieving SOAs for MANETs are introduced and discussed as in [3]. Figure 1 shows the main three parties in any SOA. A Client is supposed to find out its required service form a centralized service repository through the search phase. The repository contains a set of different offers for many service which are provided by the different service providers.

For different reasons, in a MANET, the three parties are required to be implemented together on the mobile node. The mobile node may host a service or one of its replicas, so it is considered as a "Service Provider". The same mobile host may represent a client for the hosted service, so it should considered as a "Client". Replicating the repository upon many mobile nodes is highly required in the MANET's unstable environment. So the three parties are required to be implemented in the same mobile node.

Replicable services? Which services? Composite services?

The delivered software deals with atomic services which supply specific functions and requires occupying some resources on the hosting mobile nodes. Moreover, the service proposes connected sessions for each of the requesting clients. The session holds the required information which is being processed till it is sent to the client as a response.

The service replication does not only require the executable files of the service to be replicated but also the required resources which enable this service to be functioning. A "meta" file which contains the service status and the required instructions for consistency is assumed to be included in the replication process.

Some services are not replicable by definition, for example, imagine a print out service. Here, service replication has no meaning however we do not mean installing a new printer somewhere in the network. On the other hand, we can imagine a set of replicable service like gateway service, security-key distribution services, DNS and DHCP services which can be replicated over the network participants.

About composite services, this version does not include the module of composite service execution over MANET based on SDP. This proposed module is included on an ongoing work and will be released sooner.

Service-Client Requesting Behavior (Requesting Model)

Each client has a behavior of generating the requests regarding a specific service. The number and frequency of the service requests are varying during the client lifetime.

Initially, all mobile nodes seek the initial (original) service provider node. The request frequency is the main output of the requesting model, the number of requests per a certain time interval dominates the request behavior of the clients toward a specified service. Of course, many other parameters could replace or be combined with the calling frequency like the call length. For simplicity, the proposed requesting model is based on the requesting frequency (calling rate).

Figure 2 shows how SDP models the requesting behavior –requesting model- from a client regarding a specified service, more about this model is shown in [2].

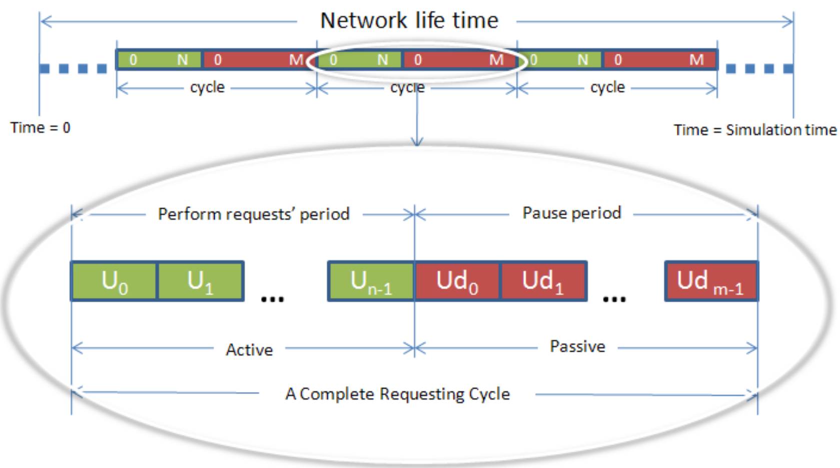


Figure 2: Requesting Model of SDP

As we can see in Figure 2, the network life time is divided into cycle. Each cycle consists of two time-length variant periods; the requests' and pause periods. In the request period the mobile client is active and perform the service requests under the influence of the calling rate while it stops of requesting the service during the pause period. M and N specify the number of allowed U and Ud units. Specifying the values and their distributions of M, N, U and Ud conform the requesting model of a client regarding a specified service.

We claim that by using such a model, we can model whatever requesting behavior. Please check [2].

Service Gross Interest

The requesting behavior of a group of clients regarding a specified service generates a common interest for this service which is the gross interest. The service replication-hibernation-restoration processes are dependent on the gross interest. [2] presents a way to quantify the gross interest and introduce different (rich/poor) gross interests based on some criteria.

Assumptions

This version of SDP-Implementation in Opnet® introduce the set of SDP core functions. The motivation behind the work is to evaluate the SDP in powerful simulation environment. Therefore, a set of assumptions are drawn here.

Repository

Services' offers are published by the provider in the available repositories in the network. A powerful protocol for service publishing and consistent offer updating is assumed to be found.

Service Discovery

Clients are supposed to search in the available repositories to find their desired offers. So, a service discovery protocol is assumed to be used by the clients.

Service Choreography

If there exists a client who is interested in a specified service, the client knows well the required inputs, outputs and their formats of this service.

Service Ranking

The different replicas of a specified service need to be ranked based on many criteria in the service repository. Request price, delay and trust represent a set of many criteria to be taken into consideration in any service ranking process. Available ranking mechanism for the different replicas is assumed to be found. All the service instances will be ranked based on some "general requirement" index.

Next ongoing versions will contain some features of the previously assumed to be found.

Minimum Components to run a SDP simulation

After Setup of Opnet® project including the MANET library. You need to include the following components in the project to be able to run a SDP simulation safely and correctly.

Radio Range - Pipeline Stage

Specifying the radio transmission range of a mobile node depends on many varying factors like the transmission power. Practically, It is hard to draw a real fixed transmission range of a wireless node. But, in simulation it is wide accepted to determine a certain fixed transmission range for the network participants.

In Opnet® Modeler® with Wireless, it is not easy to determine such a criterion. We do not consider this issue as a normal user issue. It requires some programming abilities to be modified from the delivered C pipeline file. So, for a normal user we can deliver upon your request the required custom C pipeline file and the required procedures. For the programmer who read this part more details are available in the programmer guide about that.

Important node:- The transmission range at this stage is supposed to be fixed to 75 meters (if changed, the C pipeline stage should be changed accordingly).

Wireless Domain

Specifies the characteristics of the area that you would like to deploy the network. A terrain profiles may be combined.

Mobility Config Module

Specifies the mobility model and its characteristics. Custom trajectory motion paths can be also defined. At the end of specifying your simulation environment, do not forget to associate the mobility config to your mobile nodes.

Repository Node

Only one module is required. Make sure that there is at least one and only one node of the repository module in your simulation. This node is not a real mobile node. It contributes as a virtual node in your simulation.

It represents the repository module which is supposed to be found over the mobile nodes. It performs the required functionalities of publishing, holding, updating the service/replicas' offers. It performs the required responses for the offer queries posed by the client. The statistics of SDP performance are processed, updated and stored in this module.

Moreover, it enables the user to set the specifications of the original service required to be deployed in the network.

Where to find?

From the object palette, search by the name for "SerEXE SDP global_db   (Fixed node)". Add just one instance as the only one repository node required to be available in the project.

How to specify?

Right click on the repository node and choose edit where you get the menu in Figure 3. The shown menu of attributes in Figure 3 contains a set of unused attributes (they are proposed-reserved for later usage in next versions) like (Network size and delta Time()). All time values in this menu are in seconds.

The most important settings here are Radio range in meters(here, you should specify the radio range of your mobile nodes in order to enable the repository node to compute the general performance statistics) and the “Service Distribution Analysis each” in seconds (it is the time interval of collecting the statistics about the current service distribution over the network).

The “Start Time” attribute represents in seconds the time when the repository start collecting the statistics about the service distribution. The “Stop Time” attribute set in seconds the maximum allowed time for the repository to collect the statistics about the service distribution (it can be longer than the simulation time but not shorter).

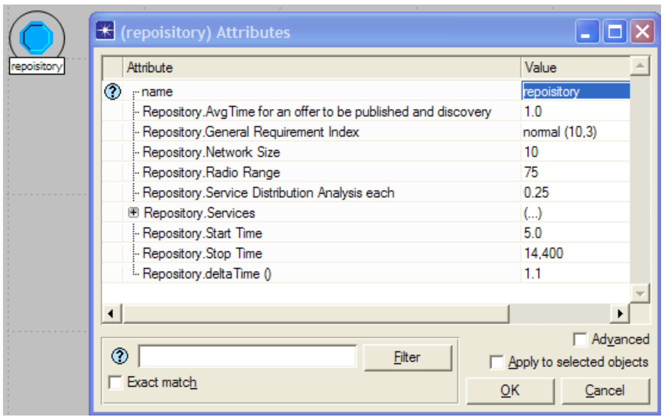


Figure 3: A repository node and its specifications

By extending the attribute of “Services”, you get the menu in Figure 4 which enables you to specify your original services in the network. For instance, in Figure 4, we have two different services; “Serv123” and “ServABC”. Initially, the two services will be deployed on a mobile node.

In order to deploy a service or one of its replica at any mobile node, it should facilitate the required disk and memory spaces to be run. The life time of a request-session (a session holds the information of the query till the processing ends at the provider side and a response is generated) is an output of the given distributions.

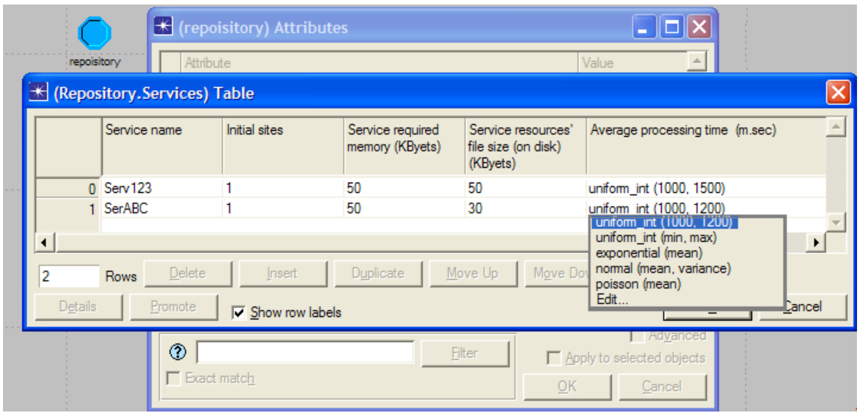


Figure 4: Specifications of the different services at the repository node

Wireless Mobile Node

The mobile node presents a network participant which may be a client or provider for the service. This node is built on the basis of an Opnet® delivered library of the wireless workstation of "manet_station_adv (Mobile node)" module. The modifications there are introduced in the programmer guide in this documentation.

Where to find?

From the object palette, search by the name for "ServExe SDP node arch (Mobile node)". Add as many instances as you want inside your wireless domain. Associate the mobility to the deployed mobile nodes.

How to specify?

Figure 5 shows the attribute menu of the mobile node. The prefix "application." refers to some required settings for the hosted application(s) by this node which is SDP-based.

1. Service Gross Interest (Requesting Model) menu:

   As in Figure 6, multiple service interests are allowed to be entered. Each of these interests (rows) indicates the requesting behavior of the application regarding a specified service.

   The interest(s) and the original available service will be associated with respect to their order. If the number of interests is greater than the number of the original available services, then the last exceeded interest(s) will be ignored. But, if the number of interests is less than the number of the original available services, then the last interest specifications will be copied to satisfy the service definitions.

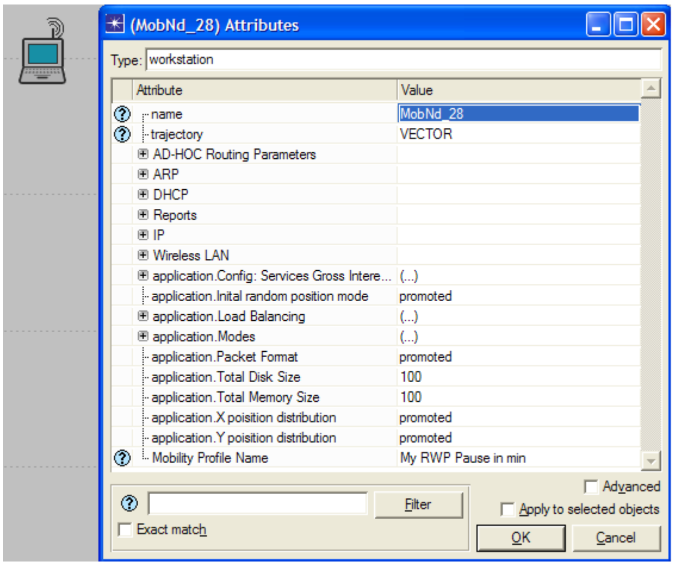


Figure 5: Specifications of the different application attributes at a mobile node

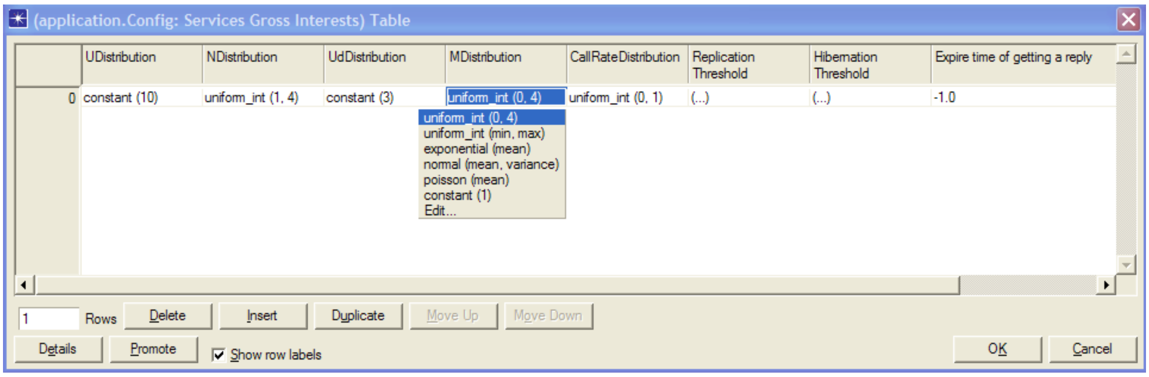


Figure 6: Specifications of the different service interests at a mobile node

2. Initial random position modes:

   This attribute is proposed-reserved for ongoing use in next versions.

3. Load Balancing

   As in Figure 7, SDP can perform a load balanced service distribution based on either the number of connected sessions at the provider side or the response time of certain request at the client side. Both of "Number of sessions" and "Response time" attribute may have -1 as a value to represent the infinity value.

Important note:- If any of the load balancing modes are turned on, then automatically the mode of "Wakeup Neighbors" of SDP [4] is turned on, too.

In this version the functioning "Load Balancing" is the "Mode: Number of sessions". The other one is proposed-reserved for ongoing use in next versions.

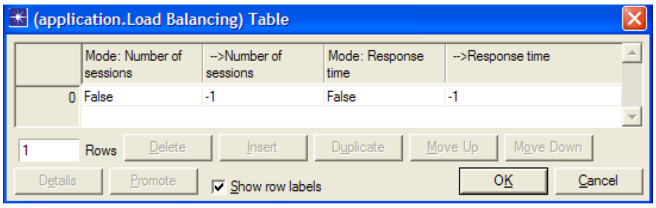


Figure 7: Specifications of the Load Balancing mode at a mobile node

4. Modes

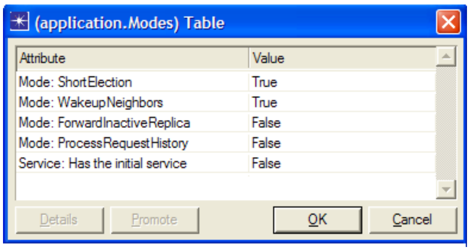


Figure 8: Specifications of SDP modes at a mobile node

In this attribute menu, shown in Figure 8, the service selection modes [4] can be specified here.

5. Packet Format

Opnet® delivered attribute. Not suggested to be used.

6. Total Disk Size and Total Memory Size:

As in Figure 5, these attributes specify the offered memory and disk space at the mobile node in KBytes.

7. X Position Distribution and Y Position Distribution

These attributes are proposed-reserved for ongoing use in next versions.

Selecting the required Statistics

As mentioned before, the main statistics for evaluating the SDP performance are stored in the repository model. You may select them by:

1. Right click on the "repository" node.

2. Choose “Choose individual DES Statistics”.
3. Navigate in the “Module Statistics” to “Repository”
4. Select your desired matrix from the delivered different categories of SDP statistics.

More about the meanings of the delivered performance statistics are mentioned in [2].

If you have more than one original service defined in your experiment, then the statistic will be show with respect of the order of these services’ definitions as mentioned in Figure 4. The service names will be replaced by an index. For example, the “Success Ratio [0]” means the success ratio of the first mentioned service, regarding in Figure 4, and Success Ratio [0]” means the success ratio of the second mentioned service and so on.

Notes on the results

Regarding the following SDP performance criteria, please be aware about the fact that these statistics are accumulators. So, their meaningful readings will be the last value not the average

- SDP-Availability.
- SDP-Success Ratio.

Do not Forget

You should find out a good set of specifications of the routing protocol which suits your scenario, otherwise you may get some meaningless results. For example, as in the supplied sample project, if you use the AODV routing protocol, you should specify correctly some parameters like the network diameter.

Part III: Programmer Guide

SDP Packets' standard

Getting a look on how the SDP application deals with the packet sending process, message structure and interrupts will be useful in understanding of the SDP packets' standard.

Sending Packet Mechanisms

SDP-Implementation in Opnet® employs a packet based information exchange simulation. There are two categories of messaging mechanisms have been used in the delivered software.

- Send-to-inside category.

  This category contains the mechanisms that enable the mobile node to send its packet stream to itself or exchanging the queries and responses with the global repository node. This mechanism depends on the Opnet® packet sending function of "op_pk_deliver(..args..)" and its derivatives like "op_pk_deliver_forced (..args..)". Review Opnet® Modeler® v.15 documentation. The current implementation considers that the global repository is supposed to be found in each mobile node, so exchanging packets here is done with the "inside" category.

- Send-to-outside category.

  The mechanisms in this category facilitate exchanging information, mainly the service requests and responses, between the network participants. Packet streams are supposed not only to be handled inside the sending participants but also through the network till reaching the destination participants. So, the function of "op_pk_send(..args..)" and its derivatives like "op_pk_sen_forced (..args..)" represent the core functionalities of these mechanisms. Review Opnet® Modeler® v.15 documentation.

SDP show an architecture independent concepts [1] which mean that it can be applied as an application processor in any node architecture of the supplied Opnet® libraries for MANET. But, it is very important to pay attention to the applied node addressing protocol in the network. The current version presents a solution which has been merged with the delivered Opnet® IPV4 and IPV6. In the SDP supplied node architecture, which has been shown in the User-Manual part of this documentation, the SDP application uses directly the auto addressing attribute of the mobile node in both mentioned packet sending categories.

Moreover, the current implementation employ the original coupling mechanism of sending packets between the application layer "traf_src" processor and the "udp" processor in the Opnet® delivered library of the wireless workstation of "manet_station_adv (Mobile node)". So, please plane for modifications in the "send-to-outside" packet sending mechanisms if you are planning to use it with another node architectures.

Message Structure

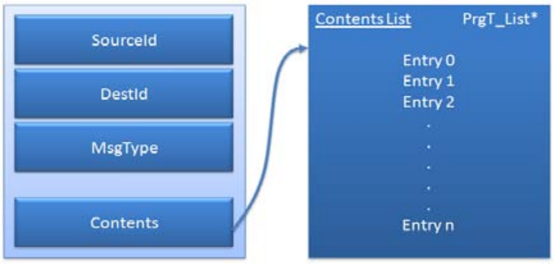


Figure 9: The general message structure of SDP.

Figure 9 shows the common structure of SDP messages which are always attached with the sent packets. The structure encapsulates the source id "SourceId", destination id "DestId", and a unique message type identifier "MsgType" with a list of a message record "Contents" which holds the different parameters regarding each message type. The "Contents" list is defined from the generic Opnet® list class "PrgT_List" which is able to hold different data types for the different entries.

Message Types and Interrupts

Based on the value of the "MsgType" identifier, Table 1 shows and describes the different message types and the involved sending and receiving parties of each of them.

MsgType	Interrupt	Sent by	Received at	Description
1	SDP_Init_Service	Repository	Host	At initialization time, the repository initiates service on the different hosts.
2	SDP_Request	Host	Host	Service query.
3	SDP_Reponse	Host	Host	Service response.
4	SDP_Request_Update	Host	Repository	Reports a(n) (un)succeeded and responded request at the repository.
5	SDP_Offer_Update	Host	Repository	Service/replica new status update control msg
6	SDP_List_Act_Rchble	Host	Repository	Lists the most interesting active replica in the local partition.
7	SDP_List_Act_Rchble_Rspns	Repository	Host	
8	SDP_List_Hib_Rchble	Host	Repository	Lists the most interesting hibernated replica in the local partition.
9	SDP_List_Hib_Rchble_Rspns	Repository	Host	
10	SDP_List_Original_Services	Host	Repository	Lists the original services Ids/names at network initialization.
11	SDP_List_Original_Services _Rspns	Repository	Host	
12	SDP_Resources_Qry	Host	Repository	A mobile asks for the required resources and "reqIndex" of a specified replica.
13	SDP_Resources_Rspns	Repository	Host	
14	SDP_Sataus_Qry	Host	Repository	A mobile node asks for the current status of a specified replica.

15	SDP_Sataus_Rspns	Repository	Host	
16	SDP_Wakeup_Rwquest	Host	Host	A “Wakeup” request from a client to a provider with an inactive replica.
17	SDP_Ask_For_Replica	Host	Host	A request to host a replica
18	SDP_Replica_Forward	Host	Host	A packet containing the replica
19	SDP_Find_next_provider	Host	Repository	A request for the Repository to find the next suitable provider.
20	SDP_Find_next_provider _Rspns	Repository	Host	A response to a host with the next valid provider in the local partition.

Table 1: Different SDP message types and descriptions.

MsgType	Interrupt	Contents’ list (parameters are separated by "|")
1	SDP_Init_Service	|SerID|Name|RepID|ReqSizeDisk| ReqSizeMemory|ProcTime|ReqIndex|
2	SDP_Request	|ClientID|SerID|RepID|Time of request| +|for_quality_reasons?(T/F)|
3	SDP_Reponse	|ProviderID|SerID|RepID|Time of request|
4	SDP_Request_Update	|ClientID|SerID|RepID|HostID|Suceeded?|
5	SDP_Offer_Update	|SerID|RepID|ProviderID|NewStatus|reqIndex| +|quality_reason (T/F)|
6	SDP_List_Act_Rchble	|ClientID|ServiceID|RequestNo|
7	SDP_List_Act_Rchble_Rspns	|ClientID|ServiceID|ReplicaID|ProviderID|PartitionNo| +|RequestNo|
8	SDP_List_Hib_Rchble	|ClientID|ServiceID|RequestNo|
9	SDP_List_Hib_Rchble_Rspns	|ClientID|ServiceID|ReplicaID|ProviderID|PartitionNo |RequestNo|
10	SDP_List_Original_Services	|ClientID|
11	SDP_List_Original_Services _Rspns	|ClientID|N:No.Services|Ser1ID|Ser1Name|...... +|SerN-11ID|SerN-11Name|
12	SDP_Resources_Qry	|ClientID|ServiceID|ReplicaID|ProviderID|RequestNo|
13	SDP_Resources_Rspns	|ClientID|ServiceID|ReplicaID|ProviderID|SizeDisk| +|SizeMemory|ProcTime|ReqIndex|RequestNo|
14	SDP_Sataus_Qry	|ClientID|ServiceID|ReplicaID|ProviderID|RequestNo|
15	SDP_Sataus_Rspns	|ClientID|ServiceID|ReplicaID|ProviderID|Status| +|RequestNo|
16	SDP_Wakeup_Rwquest	|ClientID|ServiceID|ReplicaID|Wakeup for load balance|
17	SDP_Ask_For_Replica	|ClientID|ServiceID|ReplicaID|
18	SDP_Replica_Forward	|ProviderID|ServiceID|NewReplicaID|ServiceFile|
19	SDP_Find_next_provider	|ClientID|ServiceID|LastProviderID|statusOfNextProvider|
20	SDP_Find_next_provider_Rspns	ServiceID|NextProviderID|status| NextProviderID = -1, if there is no next provider

Table 2: the Contents’ list structure w.r.t. the message type and interrupt.

Table 2 presents the Contents’ list structure with respect to the message type or interrupt. Note that the abbreviation used here are as followed:

SerID = service id, Name = service name, RepID = replica id, ReqSizeDisk= required size on disk for a service to be hosted (excutable and configuration files), ReqSizeMemory = required memory size for a service to be run, ReqIndex = service requirements' index value review [1], Ser1..N = Service id, and status= status of the service provider (active/iactive), and (T/F) indicates a Boolean value.

SDP Packets

The current version uses unformatted Opnet ® style of packets which can satisfy easily the numerous number of message types and their related message structures, review Opnet® Modeler® documentations. SDP deals only with the packet streams at the application level. As mentioned before, SDP is supposed to run only on the client /provider application layers.

Repository Node

The node architecture of the repository is simply represented as a processor. As motioned, this processor is supposed to be implemented in the mobile nodes. For simplicity, as we assumed, the repository is implemented independently. Considerations about more sophisticated implementations of this processor are issued for next versions.

Repository architecture is shown in file "SerEXE SDP global_db.nd"

Process Model

The presented process model of the repository node processor in Figure 9, see file "SerEXE SDP global_db process.pr", shows the states of:

- init: which is a state for initialization of the repository node. In this state, the following procedures take place:
    1. Reading the model attributes.
    2. Initialization for the statistics handlers.
    3. Scheduling for the interrupts of the "test" state, where the network is examined and the statistics are collected.
    4. Specifying an one meter altitude for are the mobile nodes. Actually, this step is required to enable the MANET participants to achieve a line of sight communication among them.
    5. Announcing the repository ID for the network participants.

  Two transitions are available here from the "init" state. The "default" transition and the "SART" transition. Upon the condition of "START" , the process control should move to the "deploy" state. Reviewing the source code in the entry part of the "init" state can give more details about that.

- deploy: which is a state for preparing for the deploying process of the original services on the remote providers. In this state, the following interrupt is directly scheduled to trigger the service deploying process. The deploying process is done while the transition from the "deploy" state to the "idle" state takes place.

Deploying of the original service is done in order based on the available resources at the hosting mobile node. A greedy deployment algorithm is proposed there but not included in this version. So, if your scenario is resource-restrictive, you may add your own original service deployment routine at the function implementation of "initiate_original_services_on_hosts(PrgT_List* Ser_metadata)" at the process function block.

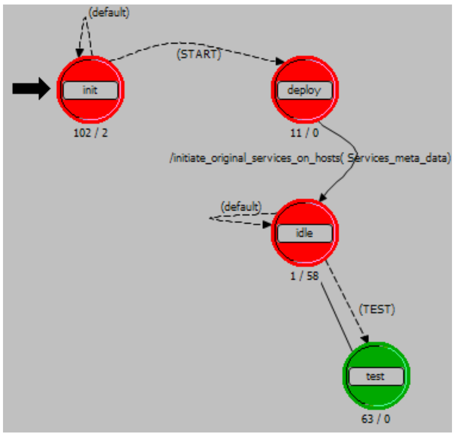


Figure 10: The repository process model.

- idle: this state has two main functions. The first important function is hosting the repository node listener in the exit part of the state. More details about the listener implementation are given sooner. The second important function is to trigger the state of "test" upon realizing the transition condition of "TEST". The required interrupts are set in the "init" state.
- test: the statistics are performed and collected here based on the given model attribute of "Service Distribution Analysis each:", review the "Manual" part of this documentation.

Repository Listener implementation

The following code portion, shown in Figure 11, presents the listener implementation of the repository node. In the first rectangle, the listener checks the inwards packet stream of the repository processor and gets the received packet if any. In the second rectangle, the listener extracts the message record of the arrived packet. In the last rectangle, based on the message type, a suitable routine will be triggered to meet the matched message interrupt then the message will be handled. Finally, after handling the message, it will be destroyed and its recourses will be released.

Statistics and Readings

As mentioned in the release notes of SDP-Implementation in Opnet®, the current number of supported original services are ten. In order to increase this number, the programmer is asked to modify the constant of “MaxNr_Original_services” at the header block of the “SerEXE SDP global_db process.pr”.

```
if (op_intrpt_type()==OPC_INTRPT_STRM)
{
    Packet* rcv_pkt_ptr = OPC_NIL;
    rcv_pkt_ptr = op_pk_get (op_intrpt_strm () );
    if (rcv_pkt_ptr != OPC_NIL)
    {
        Message_record* tmp = new Message_record();
        op_pk_fd_get (rcv_pkt_ptr, 0, &tmp);
        switch  (tmp->MsgType)
            {
                case    SDP_Request_Update  :
                    //Service_Request_Report();
                    break;
                case    5   :   //SDP_offer_update
                    Update_Service_Offer(tmp);

                    break;
                case    SDP_List_Act_Rchble :
                    List_Act_HIB_Rchble(true, tmp);
                    break;
                case    SDP_List_Hib_Rchble :
                    List_Act_HIB_Rchble(false, tmp);
                    break;
                case    SDP_List_Original_Services:
                    List_originals(tmp);
                    break;
                case    SDP_Resources_Qry   :
                    //Resources_Query();
                    break;
                case    SDP_Sataus_Qry  :
                    //Status_Query();
                    break;
                case    SDP_Find_next_provider :
                    SDP_Find_next_provider_fn(tmp);
                    break;
                default:
                    break;
            }

        delete tmp;
        op_pk_destroy (rcv_pkt_ptr);

    }
}

 intrpt_code = op_intrpt_code();
```

Figure 11: The listener implementation of the repository node.

SDP Mobile Node

In the file “ServExe SDP node arch.nd”, the node architecture of a mobile node is implemented (to be easily included in any SDP project). Figure 12 shows that architecture. The important note here to be taken into considerations is that the SDP application processor is the only processor involved by SDP processes. This architecture is the same architecture of the Opnet® Modeler® library module of manet_station_adv (Mobile node)”. The “application” processor replaced the “traf_src” processor. Review Opnet® Modeler® documentations for more details about this architecture.

SDP-Application-Fitting in the Node Architecture

Although, SDP is an architecture-independent protocol, the SDP-Implementation required some installments to be fitted in the node architecture. These installments can be concluded in the following:

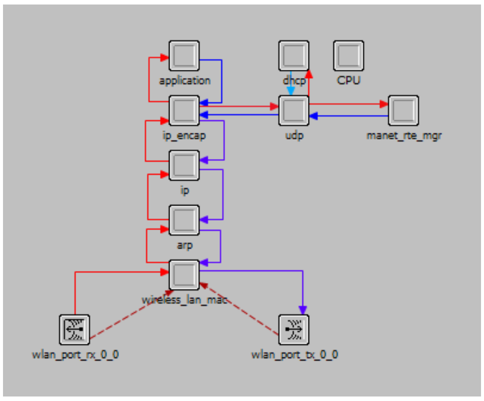


Figure 12: The SDP mobile node architecture.

- SDP uses the auto addressing mode of the mobile nodes.
- SDP uses the transportation standards in the architecture. Please review the source code of the function implementation “send_to_outside (int dest_node_id, Packet* pkt)” at the function block of the application process model at the file of “SDPrun Application.pr”.

Process model

The process model of SDP application processor is presented in Figure 13. The states are mentioned as follows:

- init: in this state, some important flags are set such as “all_events_scheduled” state variable. Both of the node “Client” and “Provider” databases (defined from class PrgT_List) are initiated with some other important lists. Both of client and provider database entries are defined in the structures of “Client_db_record_structure” and “service” respectively in the header block of the process model.
- wait, wait_0, wait1: these state organize the node address registration process for all of the network mobile nodes.
- discover: in this state, at the state enter part, the next transistion to the “wait_2 SDP_init” state is scheduled. At the exit part of the state, The IP module is discovered and registered for the current mobile node.
- wait_2 SDP_init: the function of “init_SDP()”, review the process function block, is triggered to perform:
    1. Reading the model attributes, review the “User-Manual” part of this documentation.
    2. Associating the local gross interest defined in the model attribute with the original service meta-definitions at the repository node.
    3. Starting scheduling all of the client database update, service replication and request generation interrupts.
    4. Scheduling the first hibernation check interrupt.

5. Triggering the next transition to move the process control to the “dispatch” state.

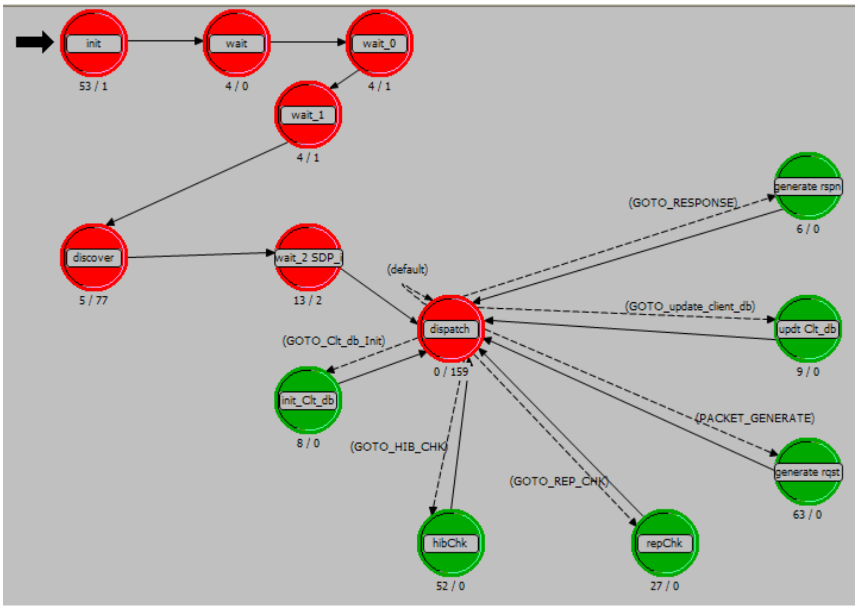


Figure 13: The process model of a SDP application.

- dispatch: on the same fashion of the repository node listener, the dispatcher states presents a listener handler stage for the inwards packet streams and arrived SDP messages. Based upon the extracted message interpret, the message is handled.
- init_Clt_db: the client database list is initiated and packed with the target service providers depending on the relevant defined gross interest. An only one transition will activate the state at the beginning of the run time.
- hibChk and repChk states: are activated based on the specifications of the hibernation and replication thresholds taken into consideration the service selection modes.
- generate_rqst: activated when the time of a request for a specific service is now.
- update_Clt_db : activated periodically (before a request generation) to keep the client database list updated about the most interesting services or replicas which are available and reached.
- generate_rspns: for a provider mobile node, the service requests are received regarding a specific offered service. The provider prepares a response for the arrived service request. To achieve the required queries in the service request, a time interval-request-session is required to hold the response information till it is sent. This state is called when one of the current request sessions has a time out and the response is ready to be sent.

Interactions among the Mobile nodes and the Repository

Service initiation process

The meta information about the given original services are defined as shown in the repository's attributes. Each of the defined services has a number of initial sites. At the beginning of the simulation the repository node performs a test at each candidate mobile node to host a service (or one of its initial copies). Based on the required service recourses, the repository determines if the mobile node will receive a service copy or not. Listing 1 shows a pseudo code for the logic behind the service initiation deployment.

```
For  all Service: Sx
        For all of Sx initial sites: Ix
              For All Mobile Nodes: Nx
                     If  ( the resources required for Ix are available on Nx)
                     then
                          Deploy Sx on Nx as an intial site
                          Break
```

Listing 1: Pseudo code of the initial deployment of services.

Figure 14 show how the repository interacts with a mobile node during the service deploying process .*

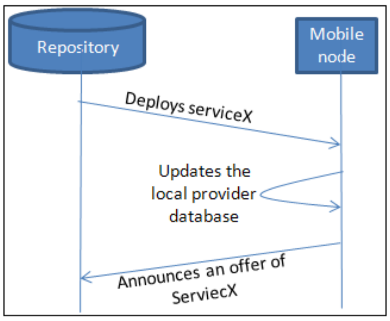


Figure 14: Interactions between the repository and a node during the service initial deployment.

Service request-response process

As mentioned in Figure 15, the service request-response process is taken place over two provider and client mobile nodes. Of course, the provider and client is supposed to be connected or at least there is a route to the client node. The shown communication lines in Figure 14 refers to streams at the service application layers.

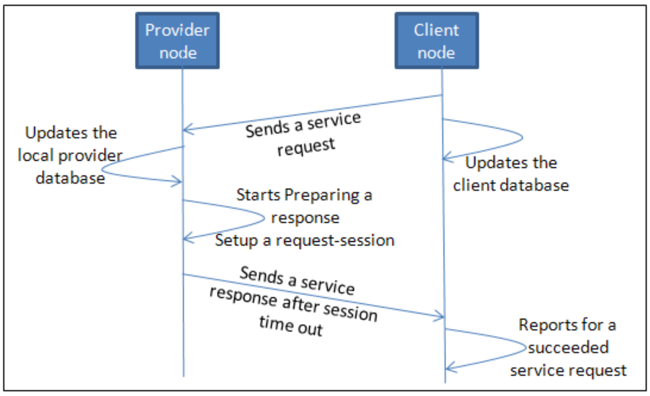


Figure 15: Interactions of a service request-response-provider-client.

Replica forward (service replication) process

The interactions in this process involve a provider and client nodes with the repository as shown in Figure 16. The client node after this process is supposed to become an active service provider.

Service hibernation process

As shown in Figure 17, an active service provider performs this process upon the low gained service interest to shut down the offered service, update the repository node and free the service occupied resources.

Service restoration process

As shown in Figure 18, an inactive service provider performs this process upon the noticed high gained service interest to activate his offered service, update the repository node and allocate the required resources.

Neighbor wakeup process (Wakeup neighbors' mode)

As shown in Figure 19, in the "Wakeup neighbor" mode, after a client database is updated, no active providers have been supplied, in this case, the client starts searching the repository for another inactive service provider in the neighborhood and to send a wakeup request to .

Forward inactive replica process (Forward Inactive replicas' mode)

As shown in Figure 20, after the client node receives his own replica of the provider node, the client is not allowed to activate the received replica. But it should update the repository database with a new offer about the inactive replica.

Load balanced service request process

Figure 21 shows a load balanced service request process in which there are two service clients. Assume the provider can only serve one client at a time. Client A posed a service request to the service provider. The provider establishes a connected session with client A. Once the session terminates, a response is sent to client A. During the session being opened for the service request of client A, another request of client B arrived. Since, the service provider, as assumed, can only serve one opened session at a time, a "busy" message will be sent to client B. In this case, client B is supposed to send a "find a next provider" to the repository. This request holds the last busy provider which the service request has been directed to.

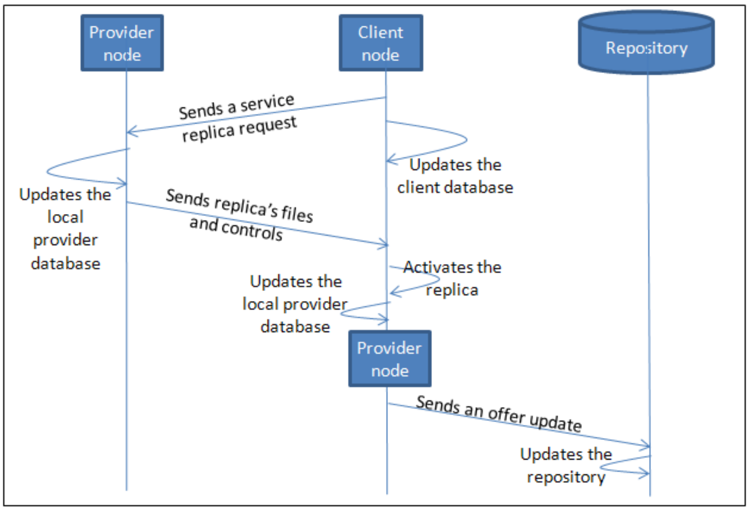


Figure 16: Interactions in a replica forward process.

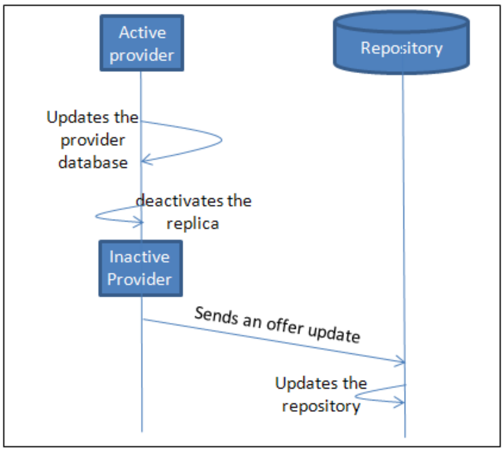


Figure 17: Interactions during a process of service hibernation.

The response of the repository contains the next active provider to the last busy provider based on the requirement index. If there is no active replica inside client B partition, the response will contain the next inactive provider to the last busy provider based on the requirement index. If there is no active or inactive providers next to the last busy provider, the , the response will contain a "-1" provider id.

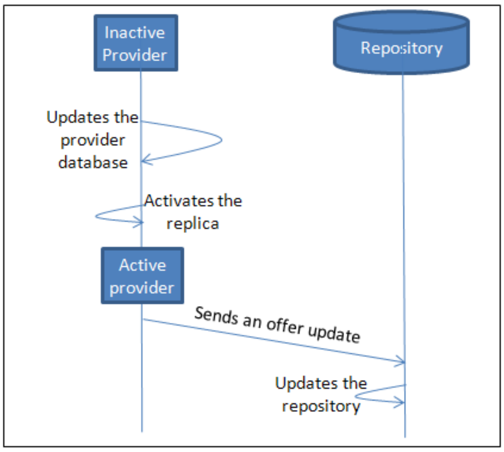


Figure 18: Interactions of a process of service restoration.

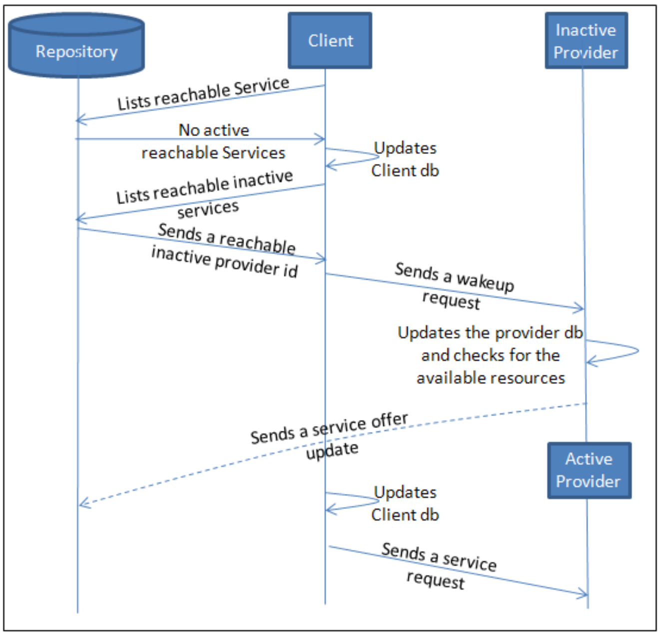


Figure 19: Interactions during a “wakeup neighbor” process.

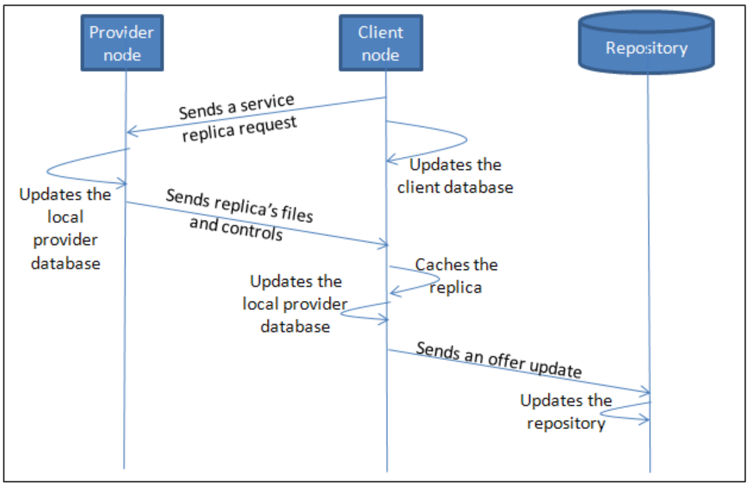


Figure 20: The interactions during a "Forward Inactive replicas" process

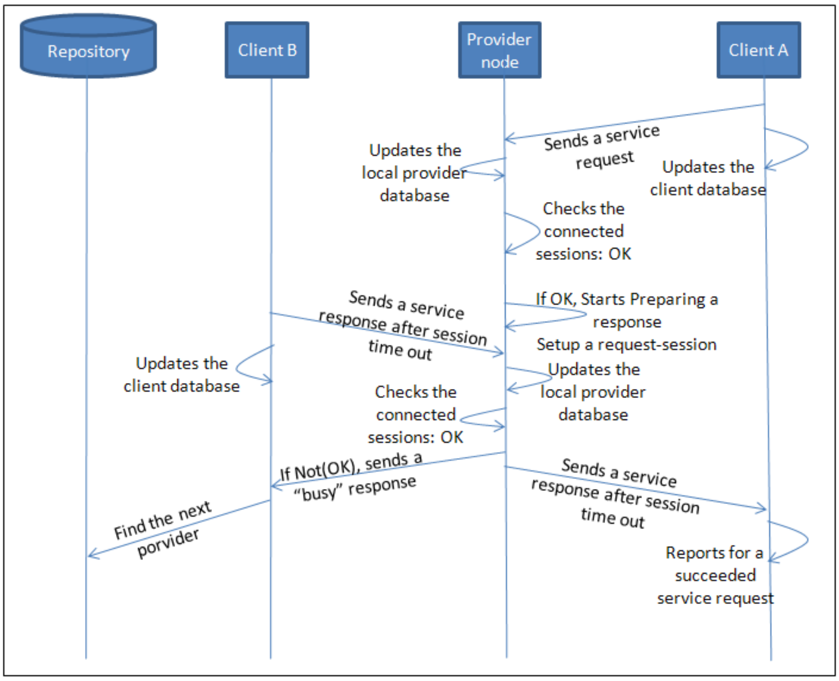


Figure 21: The interactions during a "load balanced" service request process

Service Selection Modes

The service selection modes can be set from the attribute of "Service.Modes" in the mobile node attributes. They offer a wide range of service selection mechanisms. The service selection processes are not only responsible for keeping the replication cost or prevalence as minimum as possible but also increasing the service availability in cases of disappeared active service providers such as in "Wakeup neighbor" mode.

Pipeline stage

Although, specifying the transmission range in the repository module is enough and may avoid many complications regarding dealing with the low level physical layer, we recommend applying some modification at the physical layer pipeline stage of the "chanmatch model" of the radio transmitter. The provided new pipeline stage C file, review the downloads' section in the release notes part in this documentation, prevents the transmitter of proceeding if the receiving party is away greater than a fixed length (75 meters).

Source code function prototype explanation and Lists-structures prototypes

The next section contains the most important functions and structure which are implemented in both of repository and mobile node process models. The un-mentioned functions or structures here may be test ,"to be implemented", called internally or not important to be mentioned.

Repository

1- General and topological manipulation functions

void init_partition_list(PrgT_List* net_partitions) ;

Inputs: a partition list.

Outputs: void.

Comments: initiates the partition lists ate the beginning of the simulation.

---

bool neighbors(int i, int j, double RadioRange);

Inputs: node_i and node _j ids with the radio transmission range.

Outputs: Boolean value.

Comments: True, if the nodes are neighbors .

---

void pack_sevices_meta_data(PrgT_List* Ser_meta_data);

Inputs: a list of the original services' metadata.

Outputs: void.

Comments: reads the original services' metadata specifications from the model attribute and pack then into the list.

---

void update_partition_information(PrgT_List* net_partitions, double RadioRange);

Inputs: a partition list and the radio transmission range of them.

Outputs: void.

Comments: update the partition list information regarding the current positions of the network participants.

2- Messaging Functions

void List_Act_HIB_Rchble(bool Act_Hib,int node_id, int serId, double RadioRange,PrgT_List* MyOffers,PrgT_List* net_partitions,int requestNr);

Inputs: status of the required offer, the requesting node id, the service id, transmission range of the node, the current offers ate the repository and a network partition list.

Outputs: void.

Comments: returns, as a massage to the requesting node, the most interesting (active/inactive) service in the offers database (list) and of course, regarding the requesting node partition .

---

void List_originals(Message_record* msg);

Inputs: a message record which contains the requesting node id.

Outputs: void

Comments: packs and send back a response for the requesting node with a list of the original services from their metadata.

---

void SDP_Find_next_provider_fn (Message_record* msg);

Inputs: a message record which contains the requesting node id.

Outputs: void.

Comments: sends pack to the requesting node a message with the next available provider (active/inactive). It sends a "-1-provider-id" if there is no (active/inactive) providers in the requesting node partition.

---

void Update_Service_Offer(Message_record* msg);

Inputs: a message record which contains the requesting node id.

Outputs: void.

Comments: unpacks the sent message record and initiate/update an new/old service offer.

3- Performance matrix functions

void compute_statisticsGrIGrIII (PrgT_List* net_partitions);

Inputs: a partition list .

Outputs: void.

Comments: computes and write on the statistics' handlers the service distribution processes performance of SDP for all services.

4- Important lists and structures prototypes

```
PrgT_List*   Offers;                 //list of the offered services and their information.
PrgT_List*   Partitions;             //holds the list of nodes' ids and their partitions' ids.
PrgT_List*   \Services_meta_data;    //stores the original service specifications
struct   offers_list_entry;          // "Offers" list entry
struct   Message_record;             //message record structure to be attached to the packets
struct   Partition_list_entry;       // "Partitions" list entry
struct   Service_metadata_entry;     // "Services_meta_data" list entry
```

Mobile Node

1- Initialization and tests' functions

void ask_for_a_replica(int ServiceID, int ReplicaID, int provider_id);

Inputs: the desired service id , replica id, and the hosting provider id.

Outputs: void.

Comments: composes a request for the passed service and replica ids.

---

bool available_resources(int ServiceID);

Inputs: a service id.

Outputs: Boolean .

Comments: Searchs in the service metadata at the repository database for the required resources of a specified service and compare it to the available service of the current node. Returns true if the resources are available.

---

bool chk_hibernate(int ServiceID);

Inputs: a service id.

Outputs: Boolean.

Comments: if the service gained low interest (request, regarding the hibernation threshold), it returns true .

---

bool chk_replicate(int ServiceID);

Inputs: a service id.

Outputs: Boolean.

Comments: if the service gained

---

void hibernate(int serId);

Inputs: the service id.

Outputs: void.

Comments: hibernates a specific active service/replica using the service id

---

static void init_SDP();

Inputs: void.

Outputs: void.

Comments: reads the SDP attributes of the current mobile node and extract the defined service(s) interests(s) from the "gross interest" attribute and save then in the "interest_list" list. Moreover, it initiates the requesting (calling) model specifications.

---

static void initialize_client_db();

Inputs: void.

Outputs: void.

Comments: initialize the client database.

---

void schd_the_events_for_all();

Inputs: void.

Outputs: void.

Comments: schedule all interrupts/events of request generation, service replication tests, and the initial hibernation test of the current mobile node.

high interest (request, regarding the replication threshold), it returns true

---

static bool update_a_new_calling_cycle(PrgT_List* interest_list, int s_service_id, double which_time, bool &continue_schd);

Inputs: the interest list, a service id, the time of simulation, and a flage used to toggle the scheduling processing in the pause period of the calling model (with calling rates > 0 call/time unite).

Outputs: Boolean.

Comments: Based on a service interest, regarding a specified service, the calling (requesting) model parameters are updated.

2- Client and Provider databases' manipulation functions

void activate_a_local_offer(service* local_offer, bool quality_reason);

Inputs: a local offer by the current mobile node and if it ist status is toggle because of a quality reason.

Outputs: void.

Comments: activates the local offer at the local provider database and sends an update for the repository node.

Client_db_record_structure* Get_Client_db_entrs(int ServieId);

Inputs: a service id.

Outputs: a client database entry.

Comments: returns the relevant entry of a specific service from the client database.

---

double get_request_response_travel_time(Message_record* msg);

Inputs: a received message record .

Outputs: Double value (time in seconds).

Comments: compute the response time of the received message.

---

bool has_an_active_service(int ServiceID);

Inputs: a service id.

Outputs: Boolean.

Comments: returns true if the current mobile node has an active replica of a specific service.

---

service* retrieve_provider_db_entry_service(int SerId);

Inputs: a service id.

Outputs: an offer entry from the provider database.

Comments: retrieves the relevant service record from a provider database.

---

bool retrieve_status_of_a_replica_in_my_Provider_db(int SerId);

Inputs: a service id.

Outputs: Boolean.

Comments: returns the status of a specified replica from the current node provider database.

---

void update_a_local_offer(service* local_offer);

Inputs: a local offer in the provider database.

Outputs: void.

Comments: updates the local offer at the local provider database and sends an update for the repository node.

---

void update_the_local_client_db(Client_db_record_structure* target);

Inputs: a specific client database entry.

Outputs: void.

Comments: update a specific client database entry

3- General manipulation functions

int check_number_of_currently_opened_sessions_of_service(int servId);

Inputs: a service id.

Outputs: Integer.

Comments: regarding a specific service, computes the number of the opened sessions for the currently served request.

---

void increase_total_nr_of_requests(int ServiceId);

Inputs: a service id.

Outputs: void.

Comments: increases at the repository node the total number of the succeeded request (to compute the success ratio).

4- Messaging functions

bool busy_rsponse (Message_record* msg);

inputs: a received message record.

outputs: Boolean.

comments: returns true, if the message record is a "busy" response.

---

void compose_wakeupNeighbors_request_for(Message_record* rspns, bool for_quality_reason);

inputs: a received message record and a flage to indicate if the generated "wakeup" request is done because a quality reason.

outputs: void.

comments: a wakeup request packet will be generated for the inactive provider id which is contained in the received message record.

---

void indicate_the_offer_for_quality(int serid,int repid);

inputs: a service id and replica id.

outputs: void.

comments: performs the required updated after indicating that the local offer was activated for a quality reason.

---

void Init_Service_Received(int GlobalDBModule,PrgT_List* Contents, PrgT_List* ProviderDB), inputs: the repository node id, the contents of a message record and the provider database.

outputs: void.

comments: creates a service and save it in the provider db and sends a SDP_offer_update message to the global db.

---

void List_Act_Rchble_rspns(Message_record* msg);

inputs:a message record.

outputs: void.

comments: unpacks the received packet parameters and searchs for and store/update the service-replica-provider record in the client database

---

void List_originals();

inputs: void.

outputs: void.

comments: makes a query for the original services at the repository.

---

void List_originals_Rspns(Message_record* msg);

inputs: a message record.

outputs: void.

comments: associates (just with services' ids) the gross interests to the given services. if the number of services > the number of gross interests, it produces an equivalent number of gross interests to meet the number of services. The last gross interest will be used to be copied for the list of original services ids.

---

void manage_find_next_provider_for_load_balance(int ServiceId, int lastProviderId);

inputs: a service id and the last provider id, review the load balanced service request process.

outputs: void.

comments: sens a "find next provider" request for the repository node.

---

static void manage_sending_a_replica (int dest_node_id, Packet* ReplicaPkt);

inputs: a destination mobile node id and a packet containing a replica.

outputs: void.

comments: manages sending the replica packet to its destination node.

---

static void manage_sending_a_service_requets(int SerId, int replicaId, int dest_node_id, bool for_quality_reason);

inputs: a desired service id, replica id, destination node and a flag for the quality reason.

outputs: void.

comments: manages sends a service request to a specific service.

---

static void manage_sending_a_service_rsponse(Message_record* tmp_rkrd);

inputs: a message record of a service response.

outputs: void.

comments: manages sends a service response to its destination.

Packet* prepare_a_service_request_packet(int dest_id, int ServiceId, int RepId, bool for_quality_reasons);

inputs: a destination node id, a service id, a replica id and a flag for the quality reason.

outputs: a packet.

comments: generates a service request packet to be sent to a provider node.

Packet* prepare_a_service_response_packet(Message_record* tmp_rkrd);

inputs: a message record.

outputs: a packet.

comments: generates a service response packet to be sent to a provider node.

---

void SDP_A_Replica_rcvd(Message_record* rcvdReplica);

inputs: a received message record.

outputs: void.

comments: triggers the procedures of a receiving replica file.

---

void SDP_Ask_For_Replica_rcvd(Message_record* tmp);

inputs: a message record.

outputs: void.

comments: triggers the procedures of a receiving a request for a replica to be hosted by the sending node.

---

void SDP_Find_next_provider_rspns_procedure(Message_record* tmp);

inputs: a message record.

outputs: void.

comments: starts the unpack the received message record and test if to (a)sends a message request, (b)sends a "wakeup request" or (c) halt with the suggested received provider id.

---

void SDP_Wakeup_Request_rcvd(Message_record* rqst);

inputs: a message record.

outputs: void.

comments: activates the received required service based on the available resources.

---

void Send_a_service_rsponse();

inputs: void.

outputs: void.

comments: examines the opened session and sends the next scheduled response to its destination:

---

void Service_Request_Rcvd(Message_record* msg);

inputs: a message record.

outputs: void.

comments: starts the procedure of sending a service response. Opens the request-response sessions

---

void Service_Response_Rcvd(int ServiceID, double request_response_travel_time);

inputs: a service id and time in seconds

outputs: void.

comments: if the received response is valid (with time less than the defined time for receiving a response in the equiv. GI) then, updates remotely the number of the succeeded request at the repository.

---

void update_client_db();

inputs: void.

outputs: void.

comments: trace the whole entries of the Client_db and update the reachable services.

---

void wakeupNeighbors_for_service_chk(int ServiceId,Client_db_record_structure* target);

inputs: a service id and a specific client database entry.

outputs: void.

comments: tests if there is a hibernated service to compose a wakeup request for. (by sending a packet for the repository). Forwards the request for the waked-up replica.

5- Important lists' and structures' prototypes

PrgT_List*	\Provider_db;	\\ the provider database list.
PrgT_List*	\Client_db;	\\ the client database list.
PrgT_List*	\Services_interests;	\\ the service gross interests' set list.
PrgT_List*	\requests_replies;	\\the opened sessions' list.

struct	service;	\\ a provider database entry.
struct	Client_db_record_structure;	\\ a client database entry.
struct	Service_Interests_entry;	\\ a gross interests' set list entry.
struct	requests_replies_entry;	\\ the opened sessions' list entry.

References

[1] Mohamed Hamdy and Birgitta König-Ries. Book of Communications in Computer and Information Science, Book of the selected papers of the ICETE 2008, volume 48 of CCIS 48, chapter: The Service Distribution Protocol for MANETs- Criteria and Performance Analysis, pages 467-479. Springer Berlin Heidelberg, 2009.

[2] Mohamed Hamdy and Birgitta König-Ries, "Service Availability, Success Ratio, Prevalence, Replica Allocation Correctness, Replication Degree, and Effects of Different Replication/Hibernation Behavior Effects of the Service Distribution Protocol for Mobile Ad Hoc Networks -A Detailed Study-," JENAER SCHRIFTEN ZUR MATHEMATIK UND INFORMATIK, Technical Report: Math/Inf/08/08, Friedrich-Schiller-University Jena, December 2008.

[3] D I A N E - Project: Services in ad hoc Networks, http://fusion.cs.uni-jena.de/DIANE/

[4] Mohamed Hamdy and Birgitta König-Ries, "New Service Selection Strategies for the Service Distribution Protocol for MANETs," the IADIS International Conference on Wireless Applications and Computing (WAC 2010), Freiburg, Germany.

APPENDIX C

SDP-Implementation in Opnet® v.2: MANETs' Service Execution(Release Notes - User Manual - Programmer Guide)

SDP-Implementation in Opnet® v.2

SDP-Based Composite Service Execution in MANETs

- Release Notes -User Manual –Programmer Guide

FRIEDRICH-SCHILLER-UNIVERSITY OF JENA

Preface

Please refer to [1] as a base for this documentation. Based on the SDP implementation in this previous work, we complement here an SDP-based composite service execution layer for MANETs.

Having an idea about the work described in [1] is highly recommended for understanding this documentation.

Table of Contents

SDP-Implementation in Opnet® v.2

- Release Notes -User Manual –Programmer Guide

Introduction

SDP (the **S**ervice **D**istribution **P**rotocol for MANETS) [1, 2] has been introduced as an approach with many innovative mechanisms to increase the mobile service availability. It is mainly based on a new technique in service replication called "replication-hibernation-restoration". In a previous implementation [1], SDP has been evaluated as a service replication approach for mobile networks. In this version, the previous work has been extended to include consideration for the evaluation of composite service execution processes [3].

Contents:

This documentation consists of 3 parts: I. Release Notes, II. User Manual and III. Programmer guide. In the first part, the composite service execution of composite service requests in MANETs is presented. Part II shows an easy user manual which introduces how to use the delivered software under the case tool of Opnet® Modeler® Wireless. The last part shows an extension for the programmer guide in [1] to make it easy to implement your special extensions.

Who is supposed to read this documentation? which part?

Please refer to the release notes of [1].

What to download? How to Install?

What to download?

Please find and download the SDP-Composite-Execution compressed model file in the Opnet contributed models at "https://enterprise1.opnet.com/tsts/4dcgi/models_searchdialog".

The model compressed file contains the following components:

- A channel matching pipeline:
  - pipeline stage: "dlvr chanmatch75m.ps.c"
- The mobile node:
  - Process model: "CompSerExe SDP Application.pr.m"
  - Node architecture: "CompServExe SDP node arch.nd.m"
- The repository node
  - Process model: "SerEXE CompositeService global_db process.pr.m"
  - Node: "SerEXE CompositeServices global_db.nd.m"

- A sample project:
  - Project1: “SerEXE SDP compositeServices 50.project”

How to install?

Please refer to the same section in the release note of version1 at [1].

Part I: Release Notes:

Please have a look in the previous version of the release notes. This version is the second release version.

It requires:

- The same as in [1].

Abilities:

The new abilities in this release are oriented to investigate the performance attributes of the composite service execution based on SDP in MANETs.

Plus the abilities of the previous release notes[1], it can:

- Simulate and investigate the SDP performance of the resource competing services.
- Investigate the composite service sequential and parallel execution plans with expressive attributing measurements of the average response time and the composite success ratio.

Limitations:

- Considering the previously mentioned limitations in [1].
- Simulating the loops in the composite service execution plans is not considered in this version and assumed to appear in next versions.

Part II: User Manual:

Basic Concepts

Please have an idea about the basic concepts from the "basic Concepts" section in PartII of the user manual in [1].

Composite Service Execution and Managing the Execution Plans in MANETs

Some required functionalities at the mobile clients cannot be delivered by only one service. These complex client requests need to be posed to many services, each of this services delivers a different functionality. In order to get such a complex functionalities, the mobile clients need to get some knowledge about the available services in their local network partitions from the service discovery components. This information are supposed to be used by the service composition components at these clients to produce the composite service requests. In this implementation, an extended execution layer is added for SDP to enable the mobile clients to manage the generating and controlling the execution plans to meet the posed composite requests. In this report, the services are categorized into atomic and composed services. The atomic services are those which can be hosted individually by one provider. The composite service requests are composed of minor (atomic) requests for the atomic services. All the involved atomic services in our implementations are replicable [2].

Minimum Components to run a SDP simulation

After a successful setup of an Opnet® project with the MANET library, you need to include the following components in the project to be able to run SDP simulations safely and correctly. Please refer to [1].

- Radio Range - Pipeline Stage
- Wireless Domain
- Mobility Config Module
- Repository Node

Wireless Mobile Node

This node is built on the basis of the wireless mobile node architecture of the user manual of [1]. Add as many instances as you want inside your wireless domain. Associate the mobility to the deployed mobile nodes.

How to specify?

Besides the configuration in [1] for the wireless mobile node in the user manual part, we extend the specifications to include the new features of composite service execution processes

As in Figure 1, the new added tab of “application.Composite Services” shapes the needs of the current mobile node of the composite services. Based on the defined service definitions in the global repository node, the current mobile node generates a set of composite service requests. The number of the atomic service requests in the composite service (composite service length) is varying through the network operation time. Each of the mobile nodes starts composing composite requests with an initial request length (using the value of the “Initial Length (Composite Service)” field) then, it generates another longer (the length is incremented using the value of the “Step (next length)(services)” field) request and so on until all of the defined services are included in the longest possible composite service request. After a periodic time interval (defined as a distribution in the “Step (next time)(seconds)” filed in seconds), the current mobile node repeats the previously mentioned composite service requests’ generation cycle. The composite service execution has two extremes of execution styles as mentioned in [3]. These two styles are the sequential and parallel execution styles. In the sequential execution style, all the atomic service requests are performed sequentially. The current mobile node in the sequential style of execution is supposed to pose the service requests for the atomic services one by one and considers the composite request failed if any step of the execution fails. On the other hand, in the parallel execution all the dissolved requests of any composite request are parallel posed to all of the atomic services at the same time. The style of execution is separately supplied for each mobile node through the field of “Style of execution”. The field of “Composite service execution?” simply bypasses the generation of any composite services if it is set to “False”. The field of “Max Execution Time” sets a time constraint (the “-1” value indicates that no constraint is assigned) for any of the generated composite service requests to be executed. If the response time of any of the generated composite requests is greater than this time (in milliseconds), then this request is considered as a failed composite request.

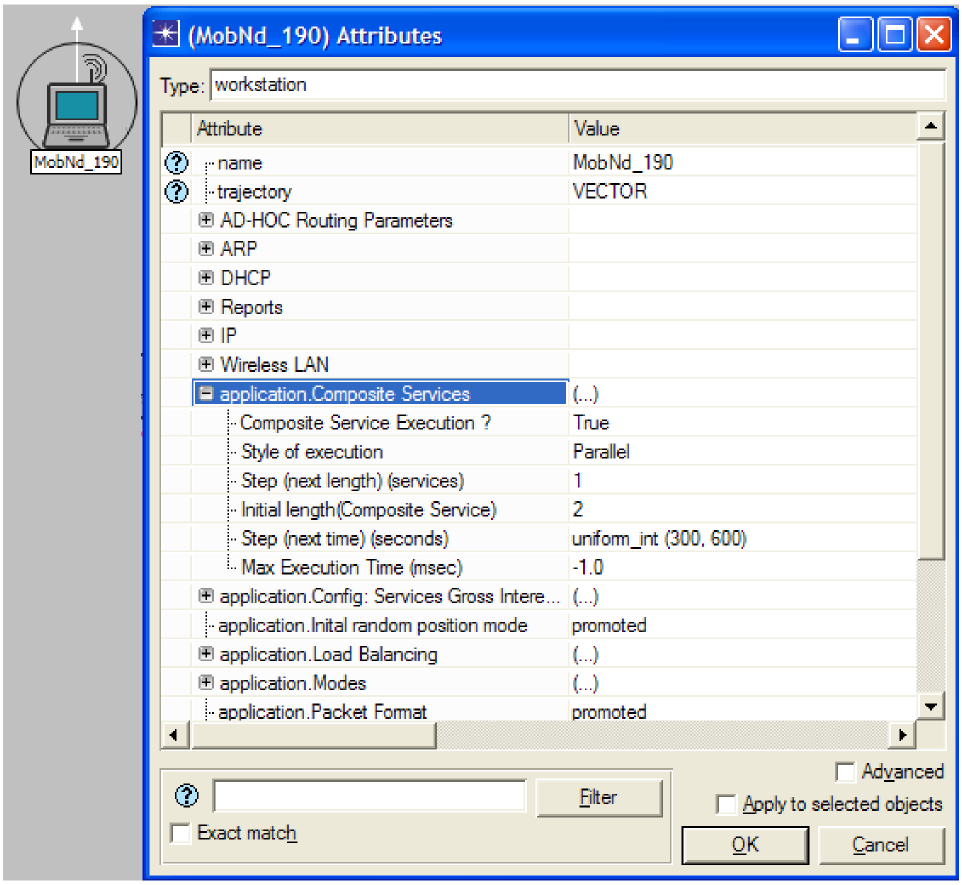


Figure 1: The mobile node fields of the process of generating the composite service requests

Selecting the required Statistics

As mentioned before, the main statistics for evaluating the SDP performance are stored in the repository model. You may select them by:

1. Right click on the “repository” node.
2. Choose “Choose individual DES Statistics”.
3. Navigate in the “Module Statistics” to “Repository”
4. Select your desired matrix from the delivered different categories of SDP statistics.

The new added statistics to indicate the performance of the new SDP-based composite service execution are contained in the tab “Composite Service”. It contains two new measurements of the composite success ratio (which reflects the ratio between the number of the succeeded composite requests of a specified length to the total number of generated composite requests from the same length) and the composite response time (which is the average of the round trip response time of the succeeded composite requests of the same length). Both of these newly added measurements are dimensional. The first index [0] of both measurements are neglected. The index of each measurement indicates the composite service "length-1".

Please refer to the user manual of [1] to get more details about the following subsections.

Notes on the results

Do not Forget

Part III: Programmer Guide

SDP Packets' standard

Getting a look at how the SDP application deals with the packet sending process, message structure and interrupts will be useful in understanding of the SDP packets' standard.

So, please refer to the previous programmer guide in version 1 in [1].

Message Types and Interrupts

The previous version of SDP implementation [1] introduced a set of messages structures for a different set of packets which present the exchanged information between the network nodes in any SDP-Opnet-project. The values of the field of "MsgType" identifier, which is defined and presented in [1], are extended in this documentation. Table 1 shows and describes the new message types and the involved sending and receiving parties of each of them.

MsgType	Interrupt	Sent by	Received at	Description
2	SDP_Request	Host	Host	Service query / Composite response.
3	SDP_Reponse	Host	Host	Service response / Composite response

Table 1: Different SDP message types and descriptions.

MsgType	Interrupt	Contents' list (parameters are separated by "|")
2	SDP_Request	|ClientID|SerID|RepID| Time of request||for_quality_reasons?(T/F)| |compositeRqst||CompositeRqstStructure|
3	SDP_Reponse	|ProviderID|SerID|RepID|Time of request|+BusyIndctr+|compositeRspns|+|compositeRqstStructure|

Table 2: the Contents' list structure w.r.t. the message type and interrupt.

Table 2 presents the Contents' list structure with respect to the message type or interrupt of the new or modified packet structures based on the previously mentioned contents' list structure in [1]. Note that the abbreviation used here are as followed:

SerID = service id, Name = service name, RepID = replica id, ReqSizeDisk= required size on disk for a service to be hosted (excutable and configuration files), ReqSizeMemory = required memory size for a service to be run, ReqIndex = service requirements' index value review [1], Ser1..N = Service id, and status= status of the service provider (active/iactive), and (T/F) indicates a Boolean value.

Composite Request Structure

A new structured field for managing the process of sending composite service requests and receiving composite responses has been added to be contained by the exchanged requests' and responses' message structures.

Figure 2 shows a configuration entry definition, in the delivered code : struct "Composite_request_configuration", which holds the execution information about both of composites request and response. Besides, holding the information about execution style, an extended descriptive record (indicator entry, in the delivered code: struct "list_indicators_entry") about the current status of the execution plan which is usually better to be kept on the requester side since, the delivered SDP-based execution implementation is centralized(see [3]). Once, any of the atomic-composite service responses arrived at its original requester, the related indicator entry of this request should be found based on the value of the indicator id field. If all of the composite service responses arrived (or a time out detected),the related indicator entry will be deleted at the requester side.

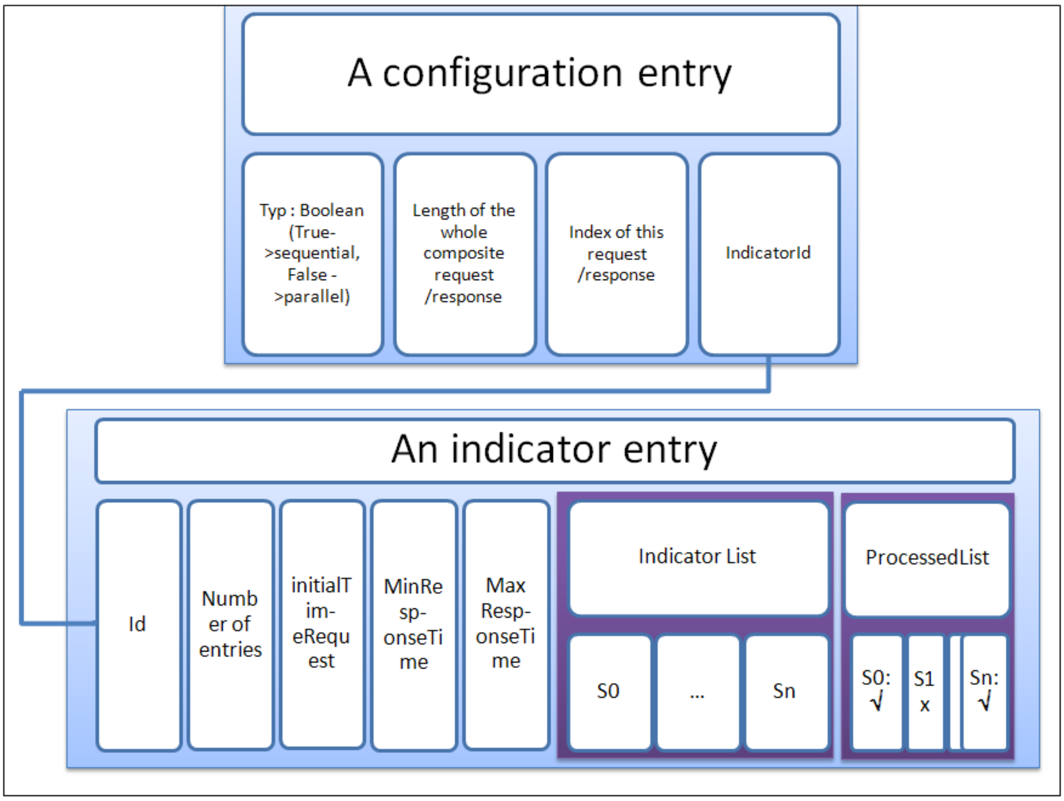


Figure 2: A configuration entry with its related indicator entry

SDP-based Composite Service Execution Messages

The current version of SDP-based execution is using the set of introduced packet formats of [1] with minor modifications for request-response message structures. The following listings 1 and 2 show the different logic behind generating and sending the messages of both sequential and parallel composite service requests.

```
for (L = 2;   L ≤ MaxNrDefinedServices;   L = L+Step)
   empty CompositeRequest_i
   for (J = 1;   J ≤ L;   J++)
      intiate(CompositeRequest_i);
      send_request(Service(J));
      wait
      until arrived_response_form(response_j,Service(J));
            proceed;
      otherwise
            break;
```

Listing 1: The logic in the sequential composite service request generation

```
for (L = 2;   L ≤ MaxNrDefinedServices;   L = L+Step)
   empty CompositeRequest_i
   for (J = 1;   J ≤ L;   J++)
      intiate(CompositeRequest_i);
      include(CompositeRequest_i,Service(J));
      send(CompositeRequest_i);
```

Listing 2: The logic in the parallel composite service request generation

The following components and process are mentioned in details in [1].

- Repository Node
- Process Model
- Repository Listener implementation
- Statistics and Readings

SDP Mobile Node

In the file "CompServExe SDP node arch.nd.m", the node architecture of a mobile node is implemented (to be easily included in any SDP project). It is the same mobile node architecture of [1]. Please refer to the documentation of [1] to have more details about the implementation of fitting a SDP-application in the proposed node architecture.

Process model

Based on the processes model which has been represented in [1], the process model of SDP application processor is slightly modified. As shown in Figure 3, two new states have been introduced. These new states are mentioned as follows:

- CompRqst: based on the specification of the composite service requesting behavior which have been introduced in the "How to specify" subsection in the user manual, the mobile node schedules the invocations of this state. Based on the style of the composite service execution, this state calls and controls the flow of generating the requests to the atomic services which are included by a composite service request.
- TestComp: reserved to be used in further implementations.

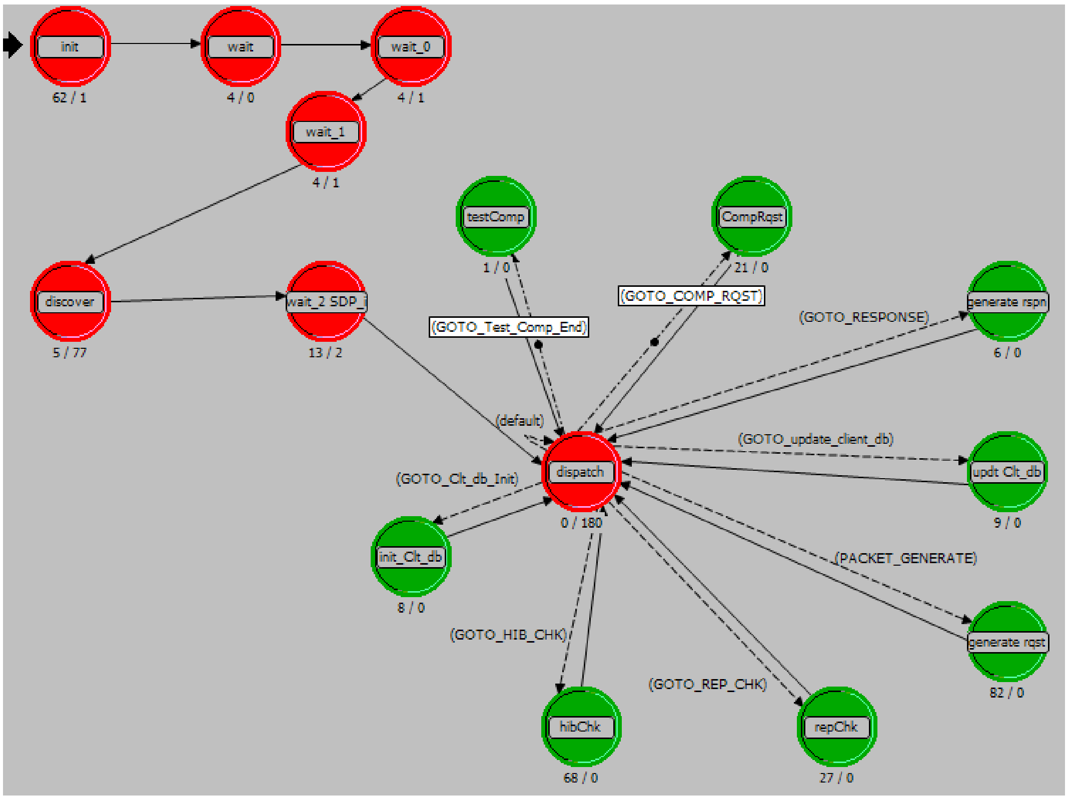


Figure 3: The new process model of a SDP application for composite service execution.

Interactions among the Mobile nodes and the Repository

Please refer to [1] for more details about the following processes and components:

- Service initiation process
- Service request-response process
- Replica forward (service replication) process
- Service hibernation process
- Service restoration process
- Neighbor wakeup process (Wakeup neighbors' mode)
- Forward inactive replica process (Forward Inactive replicas' mode)
- Load balanced service request process
- Service Selection Modes
- Pipeline stage

Composite Service Execution Styles

The composite service execution process organizes and controls the tasks of composite requests and responses for the atomic deployed services. As mentioned before, the current implementation of a SDP-based service execution aims to draw the boundaries of composite service execution based on SDP. The performance of any composite service execution will vary between two extremes of (namely: "Sequenatial" and "Parallel") execution styles. The current implementation considers a central coordination for controlling the whole execution processes where each of the clients (who issue the composite service requests) is supposed to look after its own requests for the atomic services and organize the results through the different execution steps as shown in Figure 4.

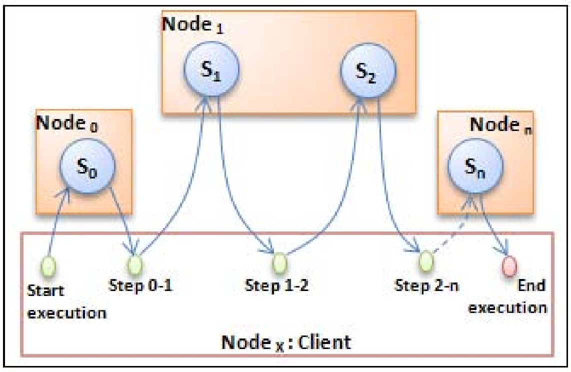


Figure 4: Centralized composite service execution

Sequential Execution Style

As shown in Figure 5-A, the client starts executing the composite service request of length n (the number of the atomic services which have been included in the request is n) by sending the first service request to the first service provider at time T=0 then, after the results of the service S0 arrived, the second request will be sent to the next service provider and so on until the end of the composite request. If there is any applied time constraint then, the execution flow should be examined during the drawn execution steps .

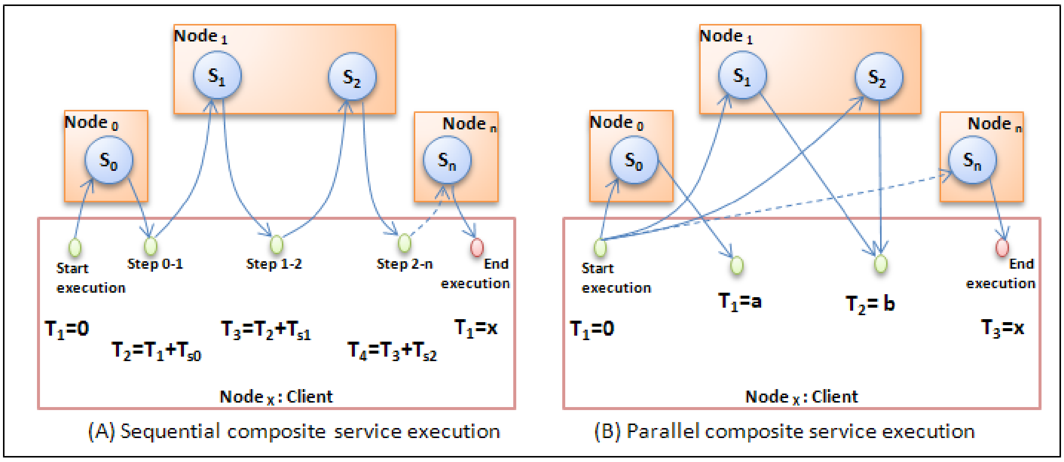


Figure 5: Sequential and parallel execution flow and control of composite service execution

Parallel Execution Style

As shown in Figure 5-B, the client starts executing the composite service request by sending to all the atomic service which have been included by the composite request at once. The execution steps here is just a set of time indications of the atomic service results' arrivals at the client.

Source code function prototype explanation and Lists-structures prototypes

This section contains only the added functions, structures and the important lists for the composite service execution. So, please refer to [1] to get more details about the rest of the source code.

Repository

1- Performance matrix functions

void compute_statisticsGrIGrIII (PrgT_List* net_partitions);

Inputs: a partition list .

Outputs: void.

Comments: computes and writes on the statistics' handlers the service distribution processes performance of SDP for all services.

2- Important structures' prototypes

struct Composite_service_statitics_entry; // To hold the information of probing the composite service execution performance matrix in the list of "Composite_service_statitics".

PrgT_List* \Composite_service_statitics;

Mobile Node

1- Composite Service Execution functions

void send_composite_service_requests_parallel();

Inputs: void

Outputs: void

Comments: manages the processes of generating and sending a set of parallel-style composite service requests to the atomic services. Each time this function is called, it iterates the composite request generation process. It start at the initial given length of the composite requests then it increments this length by a given step (length of services).

For example, if we have 5 {S1, S2, …, S5} service definitions at the repository and the initial composite request length is 2 services and step of 1 service then, once this function is called, it will generate 4 composite service requests as follows: the 1st (S1,S2), the 2nd (S1,S2,S3), the 3rd (S1,S2,S3,S4), the 4th (S1,S2,S3,S4) and the 5th (S1,S2,S3,S4,S5). So, the requester poses at the same time after this function is invoked the four previously mentioned composite request which require to be executed in parallel execution style.

---

void send_composite_service_requests_sequential();

Inputs: void.

Outputs: void.

Comments: the same as the "send_composite_service_requests_parallel" function but the generated composite services are required to be executed in parallel execution style.

Bool send_a_sequential_composite_request(int L, list_indicators_entry* indicator);

Input: length of a sequential-style composite service request, the list of the local indicators where information about this composite request will be saved.

Output: Boolean if the whole request is successfully sent.

Comments: sends an only one specified atomic request of length L to be executed in sequential style.

---

void receive_composite_service_response(Message_record* CompRspns);

Input: a message structure of an atomic service response.

Output: void.

Comments: to be called once the listener detects that the service response is belonging to a composite service response. This function receives any type (sequential/parallel) of the composite service responses.

---

void received_parallel_composites_response(Composite_request_configuration* config, double RoundTripTime);

Input: a configuration record and the related response time of getting its message.

Output: void.

Comments: for the parallel composite requests, it receives and initiates examining if all of the atomic responses of a composite response are received.

---

void sreceived_sequential_composites_response(Composite_request_configuration* config, double RoundTripTime);

Input: a configuration record and the related response time of getting its message.

Output: void.

Comments: for the sequential composite requests, it receives and initiates examining if all of the atomic responses of a composite response are received.

---

bool all_composite_responses_arrived(list_indicators_entry* indicator);

Input: the list of current not finished indicators at the current client.

Output: returns true if a specified composite service is executed (at certain time).

Comments: checks if all of the responses regarding a specified composite request is already collected.

2- Important lists' and structures' prototypes

PrgT_List* \List_of_Indicators; // Holds the minor composite requests' indicator for the composite service execution configuration record locally at the requesters.

struct Composite_request_configuration; //To be included in both of service requests and responses message structures.

struct list_indicators_entry;// Represents the indicator record to be saved at the "List_of_Indicators" list.

References

[1] Mohamed Hamdy and Birgitta König-Ries. SDP-Implementation in Opnet® v.1: MANETs' Service Replication and Load Balancing -Release Notes-User Manual–Programmer Guide, https://enterprise1.opnet.com/tsts/4dcgi/MODELS_FullDescription?ModelID=944, FRIEDRICH-SCHILLER-UNIVERSITY OF JENA, 2010.

[2]Mohamed Hamdy and Birgitta König-Ries. Book of Communications in Computer and Information Science, Book of the selected papers of the ICETE 2008, volume 48 of CCIS 48, chapter: The Service Distribution Protocol for MANETs- Criteria and Performance Analysis, pages 467-479. Springer Berlin Heidelberg, 2009.

[3] Chen et al, "A Dynamic Execution Path Selection Approach for Composite Services in MANETs," The 4th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM 08), Chengdu, China, 2008.